Universitetet i Oslo
Fysisk institutt

Hovedfagsoppgave

Adaptiv regulering av fly

Thomas Algarheim

31. Mai 2010

Master/prosjektoppgave 2010
FOR
Adaptive regulering.

Kongsberg Defence Systems (KDS) er et eget forretningsområde innen Kongsberg Gruppen ASA. KDS arbeider innen militær og sivil utvikling/produksjon. Av produkter kan nevnes raketter, simulatorer, romfartsprodukter m.m. KDS utvikler et lite fly. Prosessen, som beskriver ytelsen til flyet er gitt av størrelsen og formen på flykroppen, vingene og styreflatene og baseres på aerodynamiske data. Aerodynamiske data kommer fra vindtunnelmålinger - fysiske målinger og/eller beregnes vha CFT analyser. Oppførselen til flyet kontrolleres ved å regulere om rull, pitch og yaw. De aerodynamiske data varierer som følge av angrepsvinkler og hastighet og regulatoren skal ideelt sett tilpasses de varierende aerodynamiske forhold. Oppgaven går ut på å modellere flyet i Simulink, syntetisere regulatorer som holder flyet stabilt og som styrer flyet etter gitte baner. Deretter skal de viktigste aerodynamiske data identifiseres og brukes til å syntetisere en adaptiv regulator som beholder de spesifiserte egenskaper under alle aktuelle aerodynamiske forhold. Aerodynamiske data blir gitt av KDS.
KDS vil gi teoretisk bakgrunn før oppgaven startes. En god introduksjon til oppgaven er: Missile control av Åge Skullestad.

Forslag til framdrift:

1. Modeller flyet som en 6 DOF modell i Simulink. Inklusive i modellen skal være komplette kinematiske og dynamiske likninger. Euler parametere skal representere flyets rotasjon.
2. Flyet skal styres vha 3 indre (vinkel) sløyfer og 2 ytre (posisjon) sløyfer. Finn passende regulatorer for indre og ytre sløyfer. Bruk angrepsvinkelen = 5˚ ved syntetisering av regulatorene.
3. Dokumenter egenskapen flyet har til å følge baner i rommet
4. Sett deg inn i teorien til polplasseringsmetoden (adaptiv regulering) og dokumenter teorien
5. Stabiliteten til flyet varierer med angrepsvinkelen til flyet. Identifiser angrepsvinkelen til flyet vha Rekursive Least Square med forgetting factor eller en annen metode.
6. Syntetiser en adaptiv regulator som gir god stabilitet for alle angrepsvinkler, benytt polplasseringsmetoden.
7. Dersom tiden tillater: Benytt Åstrøm og GMV regulatorene til å syntetisere adaptive regulatorer.

Med hilsen

Åge Skullestad, Prof 2

Kongsberg Defence Systems

Forord

Jeg vil takke Åge Sullestad, min eksterne veileder ved Kongsberg Gruppen for god innføring i fly modellering, gode råd og veiledning underveis. Faget HIBU 5500 – Adaptiv regulering av Åge Skullestad har vært til stor hjelp i identifikasjon og regulering av den svært ulineære flymodellen.

Jeg vil også takke Oddvar Hallingstad som har gitt meg en god innføring i faget UNIK 4540 – Matematiske modellering av dynamiske systemer og som har hjulpet meg til å utlede den generelle flymodell med kvatrion representasjon.

Ikke minst må jeg takke min samboer Linn Charlotte Schjelderup som har vist stor tålmodighet under de to siste årene av min utdannelse.

Sammendrag

Denne oppgaven gikk ut på å lage en matematisk modell av et fly som hadde varierende stabilitet i forhold til angrepsvinkelen. Deretter ble det utviklet en adaptiv regulator som ga en raskere referanseforfølging enn en kommersiell regulator.

Dette ble gjort ved først å utvikle en matematisk modell med konstante aerodynamiske koeffisienter, som blir verifisert. Denne modellen utvidet man med å ha variabel aerodynamisk momentkoeffisient, slik at man kunne utvikle en adaptiv regulator på denne modellen. For å gjenkjenne modellen ble det diskutert forskjellige Rekursive Least Square metoder som viste seg å ikke fungere på den ulineære modellen. Derfor ble det utviklet en identifikasjons metode der man diskretiserte en forenklet modell av pitch-systemet kontinuerlig. Denne metoden viste seg å fungere svært godt så lenge de varierende aerodynamiske koeffisientene var riktig. For å regulere systemet ble poltildelingsmetoden brukt med varierende poler som tok utgangspunkt i det gjenkjente systemet.

Innhold

Tabeller

Figurer

1 Innledning

Metoder for å styre UAV[1] baserer seg stort sett på en linearisering av den ulinære flymodellen og deretter å lage en lineær regulator for dette arbeidspunktet. KDS[2] ønsker å se på hvordan det er å identifisere de viktigste aerodynamiske koeffisientene med Rekursive Least Square med Forgetting factor eller om det er andre metoder som er bedre. Med denne gjenkjenningen ønsker de å lage en lineær regulator som justerer seg etter det gjenkjente systemet, dermed har de en adaptiv regulator.

Hvis man tar skarpe svinger med flyet vil flyet ha stor forandring i sinne aerodynamiske koeffisienter, dette må man ta høyde for når man lager regulatoren. For ikke å få et ukontrollerbart system må man justere inn regulatoren til den mest ustabile tilstanden flyet vil komme til å ha. Dette fører til at flyet får en tregere respons for de andre mere stabile tilstandene flyet kommer borti. En måte å unngå dette er og juster inn regulatoren til den stabiliteten flyet har i det aktuelle tidsrommet. I en vanlig regulator vil ikke regulator parameterne bli forandret og dette fører til at vi ikke har optimal regulator til enhver tid. Dette fører til at man får en tregere referanse forfølging enn det flyet er i stand til å klare.

Formålet med denne oppgaven er å lage en matematisk modell av UVA der vi tar med full aerodynamikk med konstante aerodynamiske koeffisienter. Etter at denne modellen har blitt verifisert skal modellen bli utvidet med variabel aerodynamisk momentkoeffisient for pitch-systemet. Denne modellen skal vi regulere med en adaptiv regulator, som skal gi en raskere forfølging av referansen.

1.2 Kapitel inndeling

Kapitel 2: Vi gir en innføring i polplasseringsmetoden og hvordan vi finner passende regulatorpoler. Deretter gir vi en kort introduksjon på hvordan vi justerer inn en PID- regulator med Ziegler Nicels lukkede sløyfe metode

Kapitel 3: Her utledes den generelle ulineære flymodellen. Deretter vises styreprinsippet til flyet og hvordan dette påvirker de aerodynamiske likningene. Til slutt kommer vi frem til en forenklet modell for pitch-systemet. Den skal senere brukes til å lage en adaptiv regulator for pitch-systemet.

Kapitel 4: Her utvikles de tre indre regulatorene som stabiliserer roll, pitch og yaw med polplasseringsmetoden vi så på i kapitel 2. Deretter utvikles de to ytre regulatorene som styrer flyet etter horisontal og vertikal posisjon. Kommer deretter frem til BTT[3] prinsippet slik at vi kan utnytte flyets egenskaper på en bedre måte. Til slutt settes referansebane opp ut i fra punkter i rommet som flyet følger.

Kapitel 5: Her ser vi på stabiliteten til flyet med konstante aerodynamiske koeffisienter og hvilken effekt BTT prinsippet har på stabiliteten til flyet. Deretter innfører vi ny momentkoeffisient for pitch-systemet og ser på hva denne gjør for stabiliteten til flyet.

---

[1] Unmanned Aerial Vehicle
[2] Kongsberg Defence Systems, er et eget foretaksområde under KDA (Kongsberg Gruppen ASA)
[3] Bank-To-Turn

Kapitel 6: Her utledes vi poltildelingsmetoden som bruker den diskrete transferfunksjonen i pitch-systemet til å bestemme regulatorparameterne.

Kapitel 7: Vi utleder Recrusive Least Square med Random Walk, Faorgetting Factor og konstant trace. Deretter utvikler vi en metode der vi diskretiserer den forenklede modellen for pitch-systemet kontinuerlig.

Kapitel 8: Her bruker vi det vi har kommet fram til i de foregående kapitlene for å lage en adaptiv regulator. Først på en lineær modell av flyet og deretter på den ulineære modellen med den varierende aerodynamiske momentkoeffisienten

2. Bakgrunnsteori

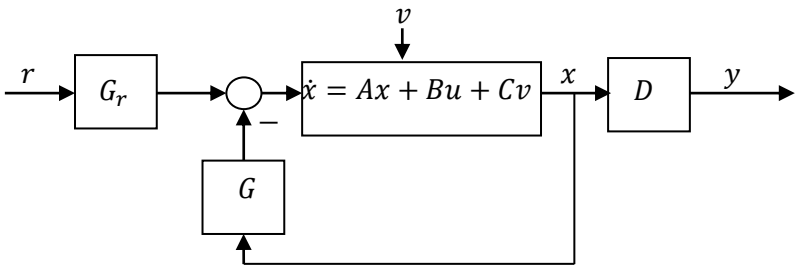
I dette kapitelet skal vi gi et teoretisk grunnlag for polplasseringsmetoden som er hentet fra (Haugen, 1996 ss. 119-160) og (Skullestad, 2007a), som skal brukes på de tre indre sløyfene i flymodellen. Vi skal også beskrive om hvordan vi bruker Ziegler Nicels metode, som er hentet fra (Haugen, 2003b ss. 96-100), for å justere inn PID regulatorer. Denne metoden skal vi bruke på de to ytre posisjonssløyfene, slik at vi skal kunne regulere den ulineære modellen via posisjon, som vi utleder i kapitel 3.

2.1 Polplassering med polynommetoden

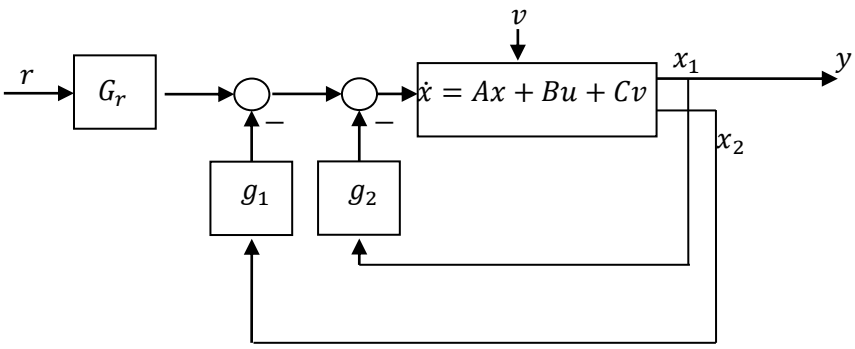
Polplasseringsregulering med tilbakekobling er en spesiell reguleringsstruktur, kjennetegnet ved at det er tilbakekobling fra hver av prosessens tilstandsvariable via forsterkninger. Dette gir oss i prinsippet muligheten til å plassere regulatorsystemets poler eller egenverdier hvor vi måtte ønske det i s-planet.

Et problem ved tilstandstilbakekopling er at det forutsetter at vi kan måle alle tilstandsvariablene. Dette problemet kan vi unngå ved å forkorte prosessen og bare se på vinkelhastighetene og stilingen til flyet.

I Matlab har vi en dedikert funksjon for å plassere egenverdiene der vi måtte ønske, denne er kalt *place,* og kan brukes til beregning av regulatorparametrene for flyet.
Denne funksjonen forutsetter parallellstruktur som vist i figur under.



Figur 1: Polplasseringsregulator på matriseform



Figur 2: Polplasseringsregulator for et system med to tilstander

Her ser vi Reguleringssystem basert på tilstandstilbakekobling med parallellstruktur. Her har vi vist hvordan vi har tilbakekobling fra alle tilstandene $n = 2$.

Vi merker oss at det er et minustegn i tilbakekoblingen.

Regulatorfunksjonen er generell

$$\begin{aligned} u &= -g_1x_1 - g_2x_2 - \cdots - g_nx_n + G_r r \\ &= -Gx + G_r r \end{aligned} \tag{2.1}$$

Der

$$G = [g_1, g_2, \dots, g_n] \tag{2.2}$$

Referanseforsterkningen G_r må generelt beregnes ut i fra følgeforholdet $\frac{y(s)}{r(s)}$ som bør ha et statisk avvik lik 1.

Vi tar utgangspunkt i regulatorsystemets følgeforhold. Vi finner følgeforholdet ved matriseberegning der vi har gitt prosessmodellen.

$$\dot{x} = Ax + Bu + Cv \tag{2.3}$$

Prosessutgangen er gitt ved

$$y = Dx \tag{2.4}$$

Siden det er følgeforholdet vi skal finne kan vi se bort i fra forstyrrelsen og setter denne til null. Regulatoren er

$$u = -Gx + G_r r \tag{2.5}$$

Setter vi 1.5 inn i 1.3 får vi

$$\begin{aligned} \dot{x} &= Ax + B(-Gx + G_r r) \\ &= (A - BG)x + Br = A_r x + B_r r \end{aligned} \tag{2.6}$$

Tar vi så Laplacetransformasjon av (2.6) og får

$$sx(s) - x_0 = A_r x(s) + B_r r(S) \tag{2.7}$$

Med $x_0 = 0$ får vi

$$x(s) = (sI - A_r)^{-1} B_r r(s) \tag{2.8}$$

Kombinerer vi (2.8) og (2.4) med følgeforholdet får vi

$$M(s) = \frac{y(s)}{r(s)} = D(sI - A_r)^{-1}B_r = D\,\frac{\text{adj}(sI - A_r)}{\det(sI - A_r)}B_r = D\,\frac{\text{adj}(sI - A + BG)}{\det(sI - A + BG)}B_r \tag{2.9}$$

Reguleringssystemets karakteristiske polynom $a(s)$ er nevnerpolynomet i følgeforholdet

$$a(s) = \det(sI - A_r) = \det(sI - A + BG) \tag{2.10}$$

Polene er røttene i $a(s)$.
Vi antar nå at reguleringssystemets poler eller egenverdier er spesifisert til å være $s_1, s_2, \dots, s_n$. Dette betyr at det karakteristiske polynomet kan skrives på formen

$$a(s) = (s - s_1)(s - s_2) \dots (s - s_n) = s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0 \tag{2.11}$$

Siden vi har polene, gir (2.10) oss ett utrykk for G og setter vi (2.10) inn i (2.11) kan vi løse ut G og finne forsterkningskonstanten i regulatoren. I praksis vil vi som nevnt over bruke Matlab funksjonen *place()* til å finne forsterkningskonstantene i regulatoren.

2.1.1 Integralvirkning

I dette tilfelle ønsker vi en regulator som har integralvirkning siden vi får påvirkninger utenifra (umodulerte tilstander) ogg med en integralvirkning vil vi få null i statisk avvik. Vi innfører derfor en ekstra tilstand x_{n+1} .

$$\dot{x}_r = \dot{x}_{n+1} = r - Dx \tag{2.12}$$

Den utvidede prosessmodellen blir som følger

$$\dot{x}_p = A_p x_p + B_p u + C_p v \tag{2.13}$$

Prosessutgangen antas gitt ved

$$y = D_p x_p \tag{2.14}$$

Den utvidede tilstandsvektoren

$$x_u = \begin{bmatrix} x_p \\ x_r \end{bmatrix} \tag{2.15}$$

Den utvidede prosessmodellen blir

$$\dot{x}_u = A_u x_u + B_u u + C_u v y = D_u x_u \tag{2.16}$$

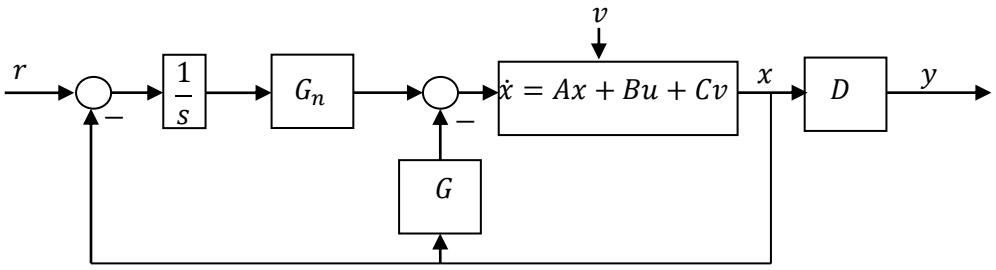
Der

$$\begin{aligned} A_u &= \begin{bmatrix} A_p & 0 \\ -D & 0 \end{bmatrix} \\ B_u &= \begin{bmatrix} B_p \\ 0 \end{bmatrix}, \qquad C_u \begin{bmatrix} C_p \\ 0 \end{bmatrix}, \qquad D_u = [D_p \quad 0\,] \end{aligned} \tag{2.17}$$

Innføring av integrator medfører normalt større følsomhet for støy og parametervariasjoner samt en kraftig økning i styresignalamplituden ved raske prosessforstyrelser. For å få en robust og ufølsom regulator med tanke på støy og parametervariasjoner plasseres polen ofte nærmere 1 enn origo. Den derved langsomme responsen på referansesignalet kan kompenseres med et nullpunkt som kan

forkortes mot nevnte pol og dermed oppheve denne virkningen. Dette får vi til med en foroverkobling.

Innføringen av integralvirkningen leder til at $G_r$faller bort og gjør det letter å finne regulator parametrene.



Figur 3: Polplasseringsregulator med integrator

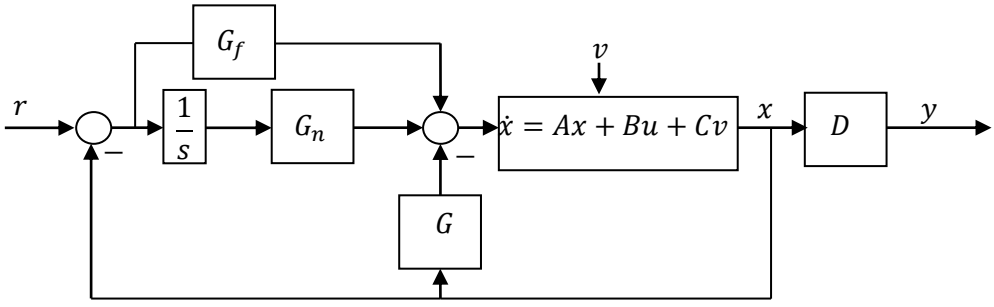
2.1.2 Foroverkobling

Siden vi ønsker ett raskt system vil vi innføre en foroverkobling som kompenserer for tregheten i integralledet og vi vil få et vesentligere raskere system.

Dimensjoneringen av tilbakekoblingene og integral skjer som før.
Integralpolen kan plasseres relativt nær 1 for å få en robust regulator mot forstyrrelse og parametervariasjoner. Foroverkoblingen velges slik at vi får et nullpunkt tilsvarende integralpolen som fører til at lukket sløyfe transferfunksjonen ikke inneholder langsomme integralvirkningen. Den beholder likevel den langsomme effekten for forstyrelser.

$$G_f = -\frac{g_{n+1}}{s_i} \tag{2.18}$$

Hvor s_i er integralpolen og g_{n+1} er integral forsterkningen.



Figur 4: Polplasseringsregulator med integralvirkning og foroverkobling

2.1.3 Hvor bør vi plassere polene?

Butterworthpolynomer

Vi kan velge regulatorsystemets karakteristiske polynom som et Butterwothpolynom. Da vil vi få et system med oversving, der Butteerworthpolynomet er gitt av

$$p_n(s)p_n(-s) = 1 + \left(\frac{s}{\omega_0}\right)^{2n} \tag{2.19}$$

Der ω_0 kalles Butterworthpolynomets knekkfrekvens, som vi kan betrakte som regulatorens båndbredde.

Sammenfallende poler

Dersom vi ikke ønsker oversving i regulatoren kan vi velge sammenfallende poler: $s_i = -\omega_0$
Det karakteristiske polynomet blir da:

$$a(s) = (s + \omega_0)^n \tag{2.20}$$

Her er ω_0 knekkfrekvensen til systemet.

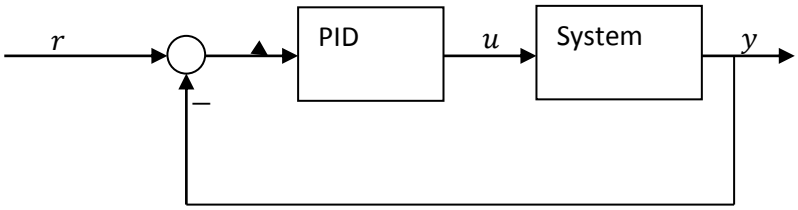
2.2 Ziegler Nicels innstiling av PID regulatorer

Denne metoden baseres på eksperimenter utført på oppkoblet regulatorsløyfe som vist under. Kravet for at denne innstillingsmetoden skal fungere er at vi kan føre systemet til stående svingninger. Dette er ikke noe problem med fly modellen.

1) Vi sørger for at prosessen er nær arbeidspunktet sitt
2) Setter $T_i = T_d = 0$ og begynner med $K_p = 0$
3) Setter på ett lite sprang i referansen
4) Øker K_p til vi får stående svingninger. Denne verdien på K_p som fører til stående svingninger, denne kaller vi den kritiske forsterkningen K_{pk}. Nå noterer vi periodetiden T_p på de stående svingningene, som er den kritiske perioden.
5) Beregner regulatorparametrene i henhold til tabellen under. Hvis vi har for stor oversving kan vi redusere K_p

	K_p	T_i	T_d
P- regulator	$0{,}5\ K_{pk}$	0	0
PI- regulator	$0.45\ K_{pk}$	$\frac{T_p}{1{,}2}$	0
PID- regulator	$0.6\ K_{pk}$	$\frac{T_p}{2}$	$\frac{T_p}{8}$

Tabell 1: Ziegler- Nichols lukkede sløyfe metode



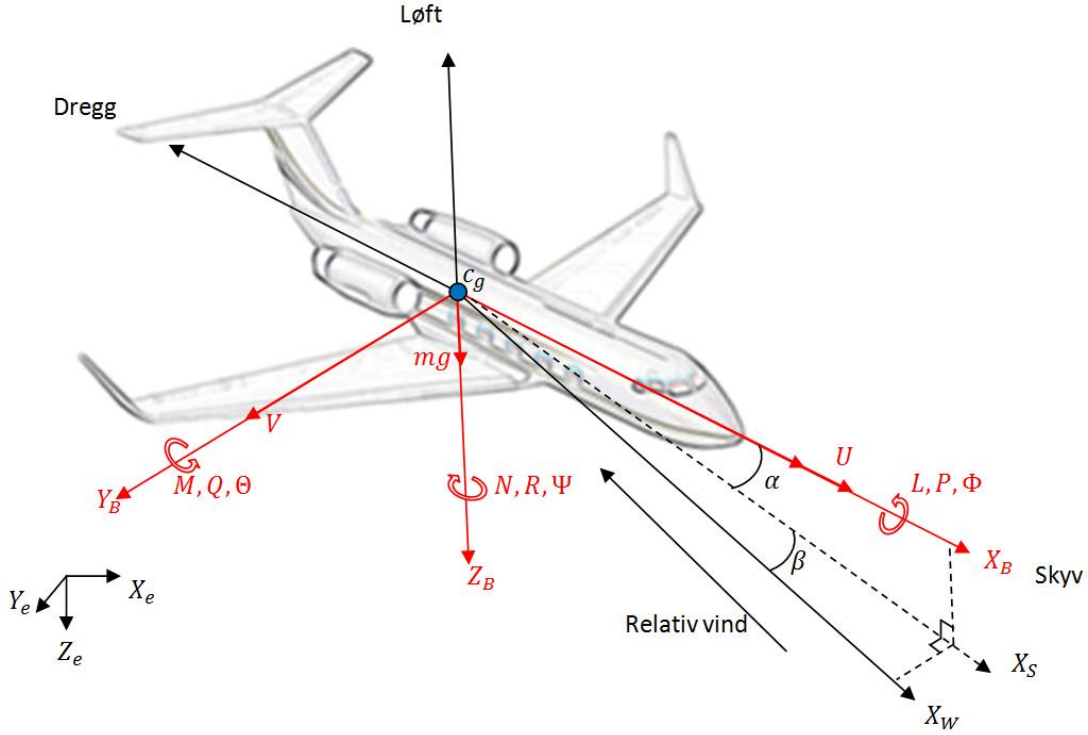
Figur 5: Oppkoblet regulatorsløyfe som Ziegler Nicels metode er basert på

3 Utledning av generell modell for fly

Vi utleder full flymodell med Eulerparametere som også kalles kvatrioner. Vi tar med de fulle aerodynamiske likningene, slik at vi får en så nær virkelig flymodell som mulig. Vi viser også likningene på hvordan vi finner de deriverte av Eulervinklene, slik at det blir lettere å komme frem til de tre indre regulatorene.

I utledningen av flyets kinematiske og dynamiske bevegelser har vi kombinert utledningene fra (McLean, 1990 ss. 16-62), (Stivens, et al., 1992 ss. 1-89), (Fossen, 1998) og (Hallingstad, 2008) de aerodynamiske kreftene er tatt fra (Stivens, et al., 1992 ss. 65-80) og (Skullestad, 2003)

3.1 Definisjon av koordinatsystemer og tilstandsvariable



Figur 6: Fly med representasjon av de tre koordinatene og de viktigste tilstandene

Kroppsaksekorset med aksene X_B, Y_B og Z_B beveger seg med flyet. X_B aksen følger lengdeaksen, Y_B-aksen går ut i høyre vinge og Z_B-aksen peker ned. Origo er tyngdepunktet til flyet, c_g.

Vindaksekorset med aksene X_W, Y_W og Z_W beveger seg med flyet. Origo er det samme som for kroppsaksekorset, dvs tyngdepunktet c_g. Som vist på figuren har vi en relativ vind fra siden og litt under. Ut i fra denne vinden er angrepsvinkel α og sideslip β definert med en plan rotasjon rundt Y_B-aksen og deretter en plan rotasjon om Z_B-aksen. I figuren over ser vi en posetiv α og β. Den første rotasjonen definerer stadilitetsaksen X_S og dermed er de andre aksene gitt.

Jordaksekorset med aksene X_e Y_e og Z_e har origo fast i punktet der flyet tok av. Jordaksekorset roterer med jorda rundt treghetsaksekorset med origo i jordas sentrum.

De viktigst variablene i utledningen av flymodellen er

$$\nu = [U, V, W, P, Q, R]^T \tag{3.1}$$

U	Hastighet i X_B- retning, langsgående hastighet
V	Hastighet i Y_B- retning, transversal hastighet
W	Hastighet i Z_B- retning, vertikal hastighet
P	Vinkelhastighet i rull, rullrate
Q	Vinkelhastighet i pitch, trimrate
R	Vinkelhastighet i yaw, kursrate

Notasjon for kvatrioner

$$\eta = [X_e, Y_e, Z_e, q_0, q_1, q_2, q_3]^T \tag{3.2}$$

X	X_e- posisjon, jordfast
Y	Y_e- posisjon, jordfast
Z	Z_e- posisjon, jordfast
q_0	Eulerparameter
q_1	Eulerparameter
q_2	Eulerparameter
q_3	Eulerparameter

Notasjon for Eulervinkler

ϕ	Rullvinkel, Eulervinkel for flyets kroppsaksekors i forhold til jordaksekorset
θ	Pitchvinkel, Eulervinkel for flyets kroppsaksekors i forhold til jordaksekorset
ψ	Yawvinkel, Eulervinkel for flyets kroppsaksekors i forhold til jordaksekorset

$$\tau = [F_X, F_Y, F_Z, \bar{L}, M, N]^T \tag{3.3}$$

F_X	Kraft i X_B- retning, langsgående kraft
F_Y	Kraft i Y_B- retning, transversal kraft
F_Z	Kraft i Z_B- retning, vertikal kraft
$\bar{L}$	Moment om X_B- akse, rullmoment
M	Moment om Y_B- akse, trimmoment
N	Moment om Z_B- akse, kursmoment

3.2 Rotasjonsmatriser mellom vindakser og stabilitetsakser (transformasjonsmatriser)

Vi trenger å finne en sammenheng mellom koordinatsystemene, dette gjør vi ved hjelp av rotasjonsmatriser. Koordinatsystemet som sitter i kroppen, B- systemet, roteres først en sideslipp vinkel β om Z_B- aksen. Det nye koordinatsystemet roteres med en angrepsvinkel α om Y_B- aksen, slik at den nye X_W-aksen peker rett mot vinden.

C_B^W er rotasjonsmatrisen fra kroppsaksesystemet, B- systemet, til vindaksekorset, W- systemet, og er gitt av Eulervinkelrepresentasjon av ortogonal retningkosinmatrisen

$$C_B^w = C_3(-\beta)C_2(\alpha) = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}$$
$$= \begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \\ -\cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \quad (3.4)$$

C_w^B er rotasjonsmatrisen fra vindaksekorset til kroppsaksekorset:

$$C_w^B = (C_B^w)^{-1} = (C_B^w)^T = \begin{bmatrix} \cos\alpha\cos\beta & -\cos\alpha\sin\beta & -\sin\alpha \\ \sin\beta & \cos\beta & 0 \\ \sin\alpha\cos\beta & -\sin\alpha\sin\beta & \cos\alpha \end{bmatrix} \quad (3.5)$$

3.3 Rotasjonsmatriser mellom kroppsaksekorset og jordakser (transformasjonsmatriser)

Bruker vi Eulervinkler får vi følgende transformasjonsmatrise mellom kroppsaksene og jordaksene.
C_e^B er transformasjonsmatrisen fra jordaksekorsett til kroppsaksekorset:

$$C_e^B = C_1(\phi)C_2(\theta)C_3(\psi)$$
$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\,\phi & sin\,\phi \\ 0 & -sin\,\phi & cos\,\phi \end{bmatrix}\begin{bmatrix} cos\,\theta & 0 & sin\,\theta \\ 0 & 1 & 0 \\ -sin\,\theta & 0 & cos\,\theta \end{bmatrix}\begin{bmatrix} cos\,\psi & sin\,\psi & 0 \\ -sin\,\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.6)$$
$$= \begin{bmatrix} cos\,\theta\,cos\,\psi & cos\,\theta\,sin\,\psi & -sin\,\theta \\ sin\,\phi\,sin\,\theta\,cos\,\psi - cos\,\phi\,sin\,\psi & sin\,\phi\,sin\,\theta\,sin\,\psi + cos\,\phi\,cos\,\psi & sin\,\phi\,cos\,\theta \\ cos\,\phi\,sin\,\theta\,cos\,\psi + sin\,\phi\,sin\,\psi & cos\,\phi\,sin\,\theta\,sin\,\psi - sin\,\phi\,cos\,\psi & cos\,\phi\,cos\,\theta \end{bmatrix}$$

C_B^e er rotasjonsmatrisen fra kroppsaksekorsett til jordaksekorset:

$$C_B^e = (C_e^B)^{-1} = (C_e^B)^T$$
$$= \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \quad (3.7)$$

Bruker vi kvatrioner får vi sammenheng mellom koordinatsystemene med en rotasjons matrise som vist i likning (2.4).

Med en rotasjon β om akse gitt av enhetsvektoren $\lambda = [\lambda_1, \lambda_2, \lambda_3]^T$ vil kvatrionene være gitt som *(Fossen 1994, Fjellstad 1994, Engeland 1996)*:

$$q = [q_0, q_1, q_2]^T = \lambda \sin\frac{\beta}{2} \quad (3.8)$$

$$q_3 = \cos\frac{\beta}{2} \quad (3.9)$$

Der enhetsvektoren $\lambda = [\lambda_1, \lambda_2, \lambda_3]^T$ er

$$\lambda = \pm\frac{q}{\sqrt{q^T q}}, \qquad \sqrt{q^T q} \neq 0 \tag{3.10}$$

Da kan vi skrive kvatrionene på følgende form

$$q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \lambda \sin\frac{\beta}{2} \\ \cos\frac{\beta}{2} \end{bmatrix}, \qquad 0 \leq \beta \leq 2\pi \tag{3.11}$$

Slik at $q^T q = 1$ eller $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$. Dette gir rotasjonsmatrisen $E_1(q)$ for lineære hastigheter fra jordaksekorset til kroppsaksekorset

$$\eta = E_1(q)v_1 \tag{3.12}$$

der

$$E_1(q) = C_E^B = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_0 q_3 + q_1 q_2) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_0 q_1 + q_2 q_3) \\ 2(q_0 q_2 + q_1 q_3) & 2(q_2 q_3 - q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{3.13}$$

Vi legger merke til at $E_1(e)$ er tronsformasjonslikningen fra jordaksekorset til kroppsaksekorsett. $E_2(e)$ er rotasjonsmatrisa for vinkelhastigheter

$$\dot{q} = E_2(q)v_2 \tag{3.14}$$

der

$$E_2(q) = \frac{1}{2}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix} \tag{3.15}$$

Dette gir oss de kinematiske bevegelseslikningene, der $\eta_E = [x, y, z\ q_0, q_1, q_2, q_3,]^T$ og $v = [U, V, W, P, Q, R]^T$.

$$\begin{bmatrix} \dot{\eta} \\ \dot{e} \end{bmatrix} = \begin{bmatrix} E_1(q) & 0_{3\times3} \\ 0_{4\times3} & E_2(q) \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \Leftrightarrow \dot{\eta}_E = E(\eta_E)v \tag{3.16}$$

3.4 Kinematiske likninger for translasjon og orientering med kvatrioner

Likning (2.6) gir oss følgende

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{q_0} \\ \dot{q_1} \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \begin{bmatrix} E_1(q) & 0_{3\times3} \\ 0_{4\times3} & E_2(q) \end{bmatrix} * \begin{bmatrix} U \\ V \\ W \\ P \\ Q \\ R \end{bmatrix} \tag{3.17}$$

Dette gir oss hastighetene i X, Y, Z sett fra jordaksekorset og endringen til kvatrionene. Utregning av likningene over gir oss følgende likninger på komponentform.

$$\dot{X} = \left(q_0^2 + q_1^2 - q_2^2 - q_3^2\right)U + 2(q_0q_3 + q_1q_2)V - 2(q_0q_2 - q_1q_3)W$$
$$\dot{Y} = -2(q_0q_3 - q_1q_2)U + \left(q_0^2 - q_1^2 + q_2^2 - q_3^2\right)V + 2(q_0q_1 + q_2q_3)W$$
$$\dot{Z} = 2(q_0q_2 + q_1q_3)U - 2(q_0q_1 - q_2q_3)V + \left(q_0^2 - q_1^2 - q_2^2 + q_3^2\right)W$$

$$\dot{q_0} = \frac{1}{2}(-q_1P - q_2Q - q_3R)\dot{q_1} = \frac{1}{2}(q_0P - q_3Q + q_2R) \tag{3.18}$$
$$\dot{q_2} = \frac{1}{2}(q_3P + q_0Q - q_1R)$$
$$\dot{q_3} = \frac{1}{2}(-q_2P + q_1Q + q_0R)$$

3.5 Orientering med Eulervinkler

Det kan være praktisk å kunne representere stilingen til flyet med Eulervinkler når vi skal linearisere flyet. Vi finner disse ved å skrive vinkelhastigheten i roll, pitch og yaw som funksjon av $\dot{\Phi}, \dot{\theta}$ og $\dot{\psi}$. Difrensiallikningene for Eulervinklene blir da

$$\begin{bmatrix} P \\ Q \\ R \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + C_1(\phi)\begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + C_1(\phi)C_2(\theta)\begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{3.19}$$

Når vi løser ut dette, med hensyn på $\dot{\Phi}, \dot{\theta}$ og $\dot{\psi}$ får vi differensiallikningen for Eulervinklene

$$\begin{aligned} \dot{\phi} &= P + \tan\theta\,(Q * \sin\phi + R * \cos\phi) \\ \dot{\theta} &= Q * \cos\phi - R * \sin\phi \\ \dot{\psi} &= \frac{Q * \sin\phi + R * \cos\phi}{\cos\theta} \end{aligned} \tag{3.20}$$

3.6 Dynamiske likninger for stivt legeme

Stivt legeme dynamikken for et fly kan skrives som

$$M_{rb}\dot{v} + C_{rb}(v)v = \tau_{rb} \tag{3.21}$$

Der

$$M_{rb} = \begin{bmatrix} mI_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_{cg} \end{bmatrix}, \qquad C_{rb}(v) = \begin{bmatrix} mS(v_1) & 0_{(3\times3)} \\ 0_{3\times3} & -S(I_{cg}v_2) \end{bmatrix} \tag{3.22}$$

Der $v_1 = [U, V, W]^T, v_2 = [P, Q, R], \tau_{rb} = [F_X, F_Y, F_Z, \bar{L}, M, N]^T$, $S(.)$ er den skjevsymmetriske matrisen og treghetsmatrisen I_{cg} er definert

$$I_{cg} = \begin{bmatrix} I_{XX} & -I_{XY} & -I_{XZ} \\ -I_{XY} & I_{YY} & -I_{YZ} \\ -I_{XZ} & -I_{YZ} & I_{ZZ} \end{bmatrix} \tag{3.23}$$

Der I_{ii}er treghetsmomenter og I_{ij} er treghetsprodukter $(i \neq j)$. Flyet er symetrisk om $X_B Z_B$ −planet slik at $I_{XY} = I_{YZ} = 0$. Kreftene og momentene som virker på flyet kan skrives som

$$\tau_{rb} = -g(\eta) + \tau \tag{3.24}$$

Der τ er en vekor som inkluderer aerodynamiske krefter og pådragskrefter. Fly modellen blir på formen

$$M_{rb}\dot{v} + C_{rb}(v)v + g(\eta) = \tau \tag{3.25}$$

På komponentform får vi

$$\begin{aligned} F_X &= m\left(\dot{U} + QW - RV + g_X^B\right) \\ F_Y &= m\left(\dot{V} + UR - WP - g_Y^B\right) \\ F_Z &= m\left(\dot{W} + VP - QU - g_Z^B\right) \\ \bar{L} &= I_{XX}\dot{P} - I_{XZ}(\dot{r} + PQ) + (I_{ZZ} - I_{YY})QR \\ M &= I_{YY}\dot{Q} + I_{XZ}(P^2 - R^2) + (I_{ZZ} - I_{YY})PR \\ N &= I_{ZZ}\dot{R} - I_{XZ}\dot{P} + (I_{ZZ} - I_{YY})PQ + I_{XZ}QR \end{aligned} \tag{3.26}$$

Hvis likningssettet (3.26) løses med hensyn på $\dot{U}, \dot{V}, \dot{W}, \dot{P}, \dot{Q}$ og $\dot{R}$, får man differensiallikningene for flyets hastigheter i X_B, Y_B og Z_B- retning, og vinkelhastigheter i rull, pitch og yaw. F_X, F_Y og F_Z er langsgående transversal og vertikale krefter

$$\dot{U} = \frac{F_X}{m} - QW + RV - g_X^B$$
$$\dot{V} = \frac{F_Y}{m} - UR + WP + g_Z^B$$
$$\dot{W} = \frac{F_Z}{m} - VP + QU + g_Z^B \tag{3.27}$$
$$\dot{P} = \frac{1}{I_{xx}I_{zz} - I_{xz}^2}\left(I_{zz}\bar{L} + I_{xz}N + \left(I_{yy}I_{zz} - I_{zz}^2 - I_{xz}^2\right)QR + I_{xz}\left(I_{xx} - I_{yy} + I_{zz}\right)PQ\right)$$
$$\dot{Q} = \frac{1}{I_{yy}}(M + I_{xz}(R^2 - P^2) + (I_{zz} - I_{xx})PR)$$
$$\dot{R} = \frac{1}{I_{xx}I_{zz} - I_{xz}^2}\left(I_{xz}\bar{L} + I_{xx}N + \left(I_{xz}^2 - I_{xx}I_{yy} + I_{xx}^2\right)PQ + I_{xz}\left(I_{yy} - I_{zz} - I_{xx}\right)QR\right)$$

Dynamikk for translasjon

$$\sum F^w = ma_{ew}^w = m^i\dot{V}_{ew}^w = m\left({}^w\dot{V}_{ew}^w + \omega_{iw}^w \times V_{ew}^w\right) \tag{3.28}$$

Flyets totale hastighet i forhold til luften er

$$V_{ew}^w = V^W + W^w \tag{3.29}$$

der $V^w = [V_T, 0,0]^T$ er totale hastigheten og W^w er atmosfærens hastighet (vind), og siden atmosfærens hastighet er uten betydning for utvikling av regulatorer og settes derfor lik null.

Vinkelhastighetsvektoren til vindaksekorset relativt til treghetsaksekorset er gitt av likningen

$$\omega_{iw}^w = \omega_{ie}^w + \omega_{ew}^w \tag{3.30}$$

der ω_{ie}^w er vinkelhastigheten til jorda som er liten (15^0/h) og kan ses bort fra.

Når dette er satt inn i Newtons 2. lov, ligning (2.8), får vi

$$\sum F^w = m\left(\begin{bmatrix}\dot{V}_T\\0\\0\end{bmatrix} + \begin{bmatrix}P_W\\Q_W\\R_W\end{bmatrix} \times \begin{bmatrix}V_T\\0\\0\end{bmatrix}\right) = m\left(\begin{bmatrix}\dot{V}_T\\0\\0\end{bmatrix} + \begin{bmatrix}0\\R_W V_T\\-Q_W V_T\end{bmatrix}\right) = m\left(\begin{bmatrix}\dot{V}_T\\R_W V_T\\-Q_W V_T\end{bmatrix}\right) \tag{3.31}$$

Kreftene som virker på flyet er skyvkraft fra jetmotoren $T^B = [, 0,0]^T$, aerodynamiske krefter (D, C, L) og tyngdekraft (mg). Blir Newtons 2. lov på følgende form når vi skriver den ut

$$T_x^w - D + mg_x^w = m\dot{V}_{\mathrm{T}}$$
$$T_y^w - C + mg_y^w = mR_W V_T \tag{3.32}$$
$$T_z^w - L + mg_z^w = -mQ_W V_T$$

Ordner vi opp i likningen over får vi

$$\dot{V}_T = \frac{1}{m}(T_x^W - D + mg_x^W)$$
$$R_W = \frac{1}{mV_T}\left(T_y^w - C + mg_y^w\right)Q_W = -\frac{1}{mV_T}(T_z^w - L + mg_z^w) \tag{3.33}$$

Flyets vinkelhastigheter som er relativt til vindaksekorset blir

$$\omega_{WB}^W = \omega_{iB}^W - \omega_{iW}^W = \omega_{ie}^W + \omega_{eB}^W - \omega_{ie}^W - \omega_{eW}^W = \omega_{eB}^W - \omega_{eW}^W$$
$$= C_B^W \begin{bmatrix} P \\ Q \\ R \end{bmatrix} - \begin{bmatrix} P_W \\ Q_W \\ R_W \end{bmatrix} \tag{3.34}$$

Vinkelhastigheten relativt vindaksekorset som funksjon av $\dot{\alpha}$ og $\dot{\beta}$.

$$\omega_{WB}^W = C_B^W \begin{bmatrix} P \\ Q \\ R \end{bmatrix} - \begin{bmatrix} P_W \\ Q_W \\ R_W \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \dot{\beta} \end{bmatrix} + C_B^W \begin{bmatrix} 0 \\ \dot{\alpha} \\ 0 \end{bmatrix} \tag{3.35}$$

$$\begin{bmatrix} P\cos(\alpha)\cos(\beta) + (Q - \dot{\alpha})\sin(\beta) + R\sin(\alpha)\cos(\beta) \\ -\mathrm{P}\cos(\alpha)\sin(\beta) + (\mathrm{Q} - \dot{\alpha})\cos(\beta) - \mathrm{R}\sin(\alpha)\sin(\beta) \\ -\mathrm{P}\sin(\alpha) + \mathrm{R}\cos(\alpha) - \dot{\beta} \end{bmatrix} = \begin{bmatrix} P_W \\ Q_W \\ R_w \end{bmatrix} \tag{3.36}$$

Setter vi dette inn i likning (3.33) og løser den med hensyn på $\dot{\alpha}$ og $\dot{\beta}$, får vi

$$\dot{\alpha} = -P\cos(\alpha)\tan(\beta) + Q - R\sin(\alpha)\tan(\beta) + \frac{1}{mV_T\cos(\beta)}(T_z^W - L + mg_z^W)$$
$$\dot{\beta} = P\sin(\alpha) - R\cos(\alpha) + \frac{1}{mV_T}\left(T_y^W - C + mg_y^W\right) \tag{3.37}$$

Dekomponering av tyngdens akselerasjon fra jordkoordinater til vindkoordinater utføres ved transformasjonene

$$\begin{bmatrix} g_x^w \\ g_y^w \\ g_z^w \end{bmatrix} = C_b^w C_e^b \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{3.38}$$

Dekomponering av motorens skyvkraft som virker i X_B- retning fra kroppskoordinater til vindkoordinater utføres ved transformasjonen

$$T^W = \begin{bmatrix} T_x^w \\ T_y^w \\ T_z^w \end{bmatrix} = C_B^w \begin{bmatrix} T \\ 0 \\ 0 \end{bmatrix} \tag{3.39}$$

3.7 Difrensiallikning for flyets bevegelser

Setter opp likningene som blir brukt i full flymodell

Total hastighet

$$\dot{V}_T = \frac{1}{m}\left(T\cos(\alpha)\cos(\beta) - D + mg_x^W\right)$$

Angrepsvinkel:

$$\dot{\alpha} = -P\cos(\alpha)\tan(\beta) + Q - R\sin(\alpha)\tan(\beta) + \frac{1}{mV_T\cos(\beta)}(T_z^W - L + mg_z^W)$$

Sideslipsvinkel:

$$\dot{\beta} = P\sin(\alpha) - R\cos(\alpha) + \frac{1}{mV_T}\left(T_y^W - C + mg_y^W\right)$$

Kvatrionene:

$$\dot{q_0} = \frac{1}{2}(-q_1P - q_2Q - q_3R)$$

$$\dot{q_1} = \frac{1}{2}(q_0P - q_3Q + q_2R)$$

$$\dot{q_2} = \frac{1}{2}(q_3P + q_0Q - q_1R)$$

$$\dot{q_3} = \frac{1}{2}(-q_2P + q_1Q + q_0R)$$

Vinkelhastighet sett fra flyet: (3.40)

$$\dot{P} = \frac{1}{I_{xx}I_{zz} - I_{xz}^2}\left(I_{zz}\bar{L} + I_{xz}N + \left(I_{yy}I_{zz} - I_{zz}^2 - I_{xz}^2\right)QR + I_{xz}\left(I_{xx} - I_{yy} + I_{zz}\right)PQ\right)$$

$$\dot{Q} = \frac{1}{I_{yy}}(M + I_{xz}(R^2 - P^2) + (I_{zz} - I_{xx})PR)$$

$$\dot{R} = \frac{1}{I_{xx}I_{zz} - I_{xz}^2}\left(I_{xz}\bar{L} + I_{xx}N + \left(I_{xz}^2 - I_{xx}I_{yy} + I_{xx}^2\right)PQ + I_{xz}\left(I_{yy} - I_{zz} - I_{xx}\right)QR\right)$$

Posisjon sett fra jordkoordinatsystemet:

$$\dot{X} = \left(q_0^2 + q_1^2 - q_2^2 - q_3^2\right)U + 2(q_0q_3 + q_1q_2)V - 2(q_0q_2 - q_1q_3)W$$

$$\dot{Y} = -2(q_0q_3 - q_1q_2)U + \left(q_0^2 - q_1^2 + q_2^2 - q_3^2\right)V + 2(q_0q_1 + q_2q_3)W$$

$$\dot{Z} = 2(q_0q_2 + q_1q_3)U - 2(q_0q_1 - q_2q_3)V + \left(q_0^2 - q_1^2 - q_2^2 + q_3^2\right)W$$

Hastigheter sett fra flyet:

$$\dot{U} = \frac{F_X}{m} - QW + RV - g_X^B$$

$$\dot{V} = \frac{F_Y}{m} - UR + WP + g_Z^B$$

$$\dot{W} = \frac{F_Z}{m} - VP + QU + g_Z^B$$

3.8 Beskrivelse av styreflater

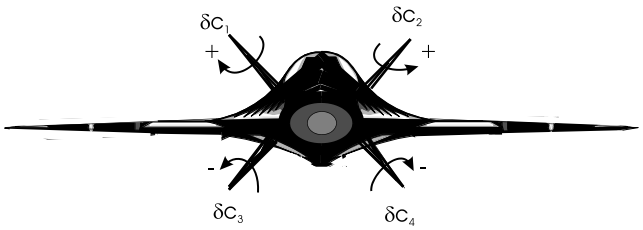
Flyet styres ved hjelp av fire styreflater c_1, c_2, c_3 og c_4 som er plassert bak vingene på flykroppen. Alle styreflatene inngår i alle tre styresløyfene på følgende måte

$$\begin{aligned} \delta R &= \delta c_1 - \delta c_2 + \delta c_3 - \delta c_4 \\ \delta P &= \delta c_1 + \delta c_2 + \delta c_3 + \delta c_4 \\ \delta Y &= -\delta c_1 + \delta c_2 + \delta c_3 - \delta c_4 \end{aligned} \tag{3.41}$$

der δR er rollvinkel, δP er pitchvinkel og δY er yawvinkel.

Som gir

$$\begin{aligned} \delta c_1 &= \delta R + \delta P - \delta Y \\ \delta c_2 &= -\delta R + \delta P + \delta Y \\ \delta c_3 &= \delta R + \delta P + \delta Y \\ \delta c_4 &= -\delta R + \delta P - \delta Y \end{aligned} \tag{3.42}$$



Figur 7: Fly med beskrivelse av rorutslag

Positiv retning for rorutslag er "baken ned", på engelsk "trailing edge down". Dette betyr at positivt utslag på alle fire ror gir negativt pitchmoment.

3.9 Aerodynamikk

Likningen for lift, drag og sidekraft er gitt av likning (3.43)

$$\begin{aligned} D &= \frac{1}{2}\bar{q}SC_D \\ C &= \frac{1}{2}\bar{q}SC_C \\ L &= \frac{1}{2}\bar{q}SC_L \end{aligned} \tag{3.43}$$

Der $\bar{q} = \rho V_T^2$ som er dynamisk trykk.

Setter vi inn C_L, C_D og C_C (Skullestad, 2003)

$$\begin{aligned} D &= \frac{1}{2}\bar{q}S(C_{D0} + C_{D\alpha 2}\alpha^2) \\ C &= \frac{1}{2}\bar{q}S\left(C_{C\beta}\beta + \frac{b}{2V_T}\left(C_{C\dot{\beta}}\dot{\beta} + C_{Cr}R\right) + C_{C\delta c_1}\delta c_1 + C_{C\delta c_2}\delta c_2 + C_{C\delta c_3}\delta c_3 \right. \\ &\qquad \left. + C_{C\delta c_4}\delta c_4\right) \\ L &= \frac{1}{2}\bar{q}S\left(C_{L\alpha}\alpha + \frac{c}{2V_T}\left(C_{L\dot{\alpha}}\dot{\alpha} + C_{LQ}Q\right) + C_{L\delta c_1}\delta c_1 + C_{L\delta c_2}\delta c_2 + C_{L\delta c_3}\delta c_3 \right. \\ &\qquad \left. + C_{L\delta c_4}\delta c_4\right) \end{aligned} \tag{3.44}$$

Der vi forenkler med å sette $C_{C\dot{\beta}}$, $C_{L\dot{\alpha}}$ lik null siden disse har liten innvirkning på totale kreftene.

Når vi skal linearisere modellen er det praktisk at vi har pådrag i Pitch, Rull og Yaw, dette får vi ved å bruke likning (3.42) på (3.44) og får

$$\begin{aligned} D &= \frac{1}{2}\bar{q}S(C_{D0} + C_{D\alpha 2}\alpha^2) \\ C &= \frac{1}{2}\bar{q}S\left(C_{C\beta}\beta + \frac{b}{2V_T}\left(C_{C\dot{\beta}}\dot{\beta} + C_{Cr}R\right) + C_{C\delta R}\delta R + C_{C\delta P}\delta P + C_{C\delta Y}\delta Y\right) \\ L &= \frac{1}{2}\bar{q}S\left(C_{L\alpha}\alpha + \frac{c}{2V_T}\left(C_{L\dot{\alpha}}\dot{\alpha} + C_{LQ}Q\right) + C_{L\delta R}\delta R + C_{L\delta P}\delta P + C_{L\delta Y}\delta Y\right) \end{aligned} \tag{3.45}$$

De aerodynamiske momentene om flyets kroppsakser er tilnærmet gitt av likningene under.

$$\begin{aligned} \bar{L} &= \frac{1}{2}\bar{q}SbC_{\bar{L}} \\ M &= \frac{1}{2}\bar{q}ScC_M \\ N &= \frac{1}{2}\bar{q}SbC_N \end{aligned} \tag{3.46}$$

Setter inn for $C_{\bar{L}}, C_M$ og C_N og får

$$\bar{L} = \frac{1}{2}\bar{q}Sb\left(C_{\bar{L}\beta}\beta + \frac{b}{2V_T}(C_{\bar{L}P}P + C_{\bar{L}R}R) + C_{\bar{L}\delta c1}\delta c_1 + C_{\bar{L}\delta c2}\delta c_2 + C_{\bar{L}\delta c3}\delta c_3 + C_{\bar{L}\delta c4}\delta c_4\right)$$

$$M = \frac{1}{2}\bar{q}Sc\left(C_{M_0} + C_{M\alpha}\alpha + \frac{c}{2V_T}\left(C_{M\dot{\alpha}}\dot{\alpha} + C_{MQ}Q\right) + C_{M\delta c_1}\delta c_1 + C_{M\delta c_2}\delta c_2 + C_{M\delta c_3}\delta c_3 + C_{M\delta c_4}\delta c_4\right) \quad (3.47)$$

$$N = \frac{1}{2}\bar{q}Sb\left(C_{N\beta}\beta + \frac{b}{2V_T}(C_{NP}P + C_{NR}R) + C_{N\delta c1}\delta c_1 + C_{N\delta c2}\delta c_2 + C_{N\delta c3}\delta c_3 + C_{N\delta c4}\delta c_4\right)$$

Som vi forenkler med å sette $C_{M\dot{\alpha}} = 0$.

Her er det også praktisk å skrive om likning (3.47) med tanke på pådrag i Pitch, Rull og Yaw, dette får vi til ved bruke likning (3.42), og vi får

$$\bar{L} = \frac{1}{2}\bar{q}Sb\left(C_{\bar{L}\beta}\beta + \frac{b}{2V_T}(C_{\bar{L}P}P + C_{\bar{L}R}R) + C_{\bar{L}\delta R}\delta R + C_{\bar{L}\delta P}\delta P + C_{\bar{L}\delta Y}\delta Y\right)$$

$$M = \frac{1}{2}\bar{q}Sc\left(C_{M_0} + C_{M\alpha}\alpha + \frac{c}{2V_T}\left(C_{M\dot{\alpha}}\dot{\alpha} + C_{MQ}Q\right) + C_{M\delta R}\delta R + C_{M\delta P}\delta P + C_{M\delta Y}\delta Y\right) \quad (3.48)$$

$$N = \frac{1}{2}\bar{q}Sb\left(C_{N\beta}\beta + \frac{b}{2V_T}(C_{NP}P + C_{NR}R) + C_{N\delta R}\delta R + C_{N\delta P}\delta P + C_{N\delta Y}\delta Y\right)$$

3.10 Forenklede systemet for pitch-systemet

Her har vi brukt (Skullestad, 2003) og (Haugen, 2003a) til å finne en forenklet pitchsystem.

Vi tar utgangspunkt i likningen (3.27) der vi ser på vinkelhastigheten

$$\dot{Q}I_{yy} = M + I_{xz}(R^2 - P^2) + (I_{zz} - I_{xx})PR \quad (3.49)$$

Der vi antar at $I_{xz}(R^2 - P^2) + (I_{zz} - I_{xx})PR$ er liten i forhold til den aerodynamiske momentet M.

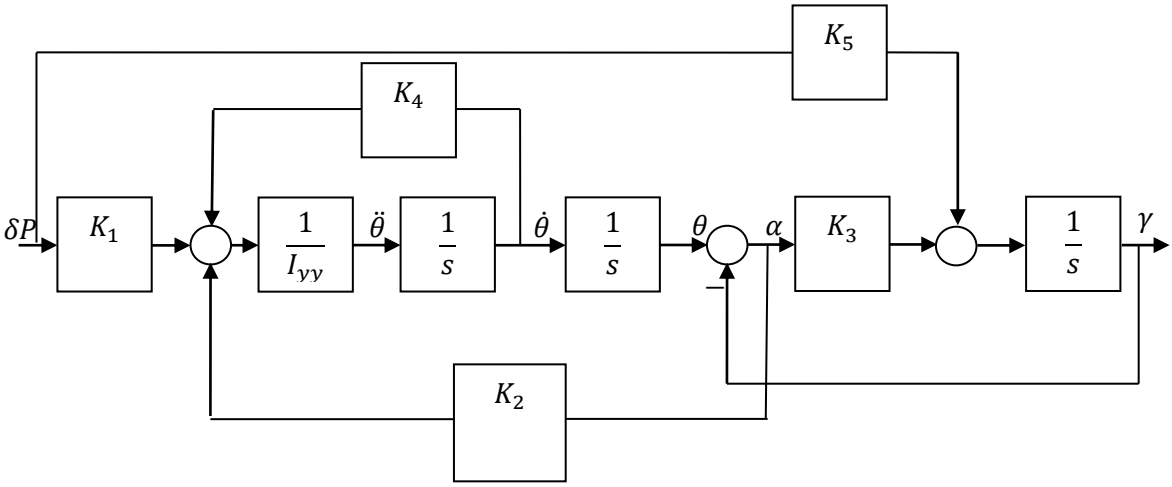
Får den tilnærmede dynamiken.

$$\dot{Q}I_{yy} = \frac{1}{2} * \rho * V_T^2 Sc\left(\frac{\partial C_M}{\partial \delta}\delta P - \frac{\partial C_M}{\partial \alpha}\alpha - \frac{\partial C_M}{\partial Q}Q\right) \quad (3.50)$$

Angrepsvinkelen kan da skrives

$$\alpha = \theta - \gamma \quad (3.51)$$

Der γ er den totale vinkelen mellom flykoordinatene og vindkoordinatene, og vi kan sette opp blokkdiagrammet for pitch-systemet



Figur 8: Blokkdiagram av forenklet Pitch-system

Der

$$
\begin{aligned}
K_1 &= \frac{1}{2}\rho V_T^2 S c C_{M\delta P} \\
K_2 &= \frac{1}{2}\rho V_T^2 S c C_{M\alpha} \\
K_3 &= \frac{1}{2m}\rho V_T S C_{L\alpha} \\
K_4 &= \frac{1}{4}\rho s V_T c^2 C_{M\omega_y} \\
K_5 &= \frac{1}{2m}\rho V_T S C_{L\delta P}
\end{aligned}
\tag{3.52}
$$

Ut i fra figuren over kan vi finne state space modellen fra

$$
\begin{aligned}
\dot{\gamma} &= K_3(\theta - \gamma) + \mathrm{K_5}\delta\mathrm{P} \\
\theta &= \dot{\theta} \\
\ddot{\theta} &= \frac{K_2(\theta - \gamma) + K_4\dot{\theta} + K_1\delta P}{I_{yy}}
\end{aligned}
\tag{3.53}
$$

med $\underline{x} = [\gamma \quad \theta \quad \dot{\theta}]^T$

$$
A = \begin{bmatrix} -K_3 & K_3 & 0 \\ 0 & 0 & 1 \\ -\frac{K_2}{I_{yy}} & \frac{K_2}{I_{yy}} & \frac{K_4}{I_{yy}} \end{bmatrix}, B = \begin{bmatrix} K_5 \\ 0 \\ \frac{K_1}{I_{yy}} \end{bmatrix}
\tag{3.54}
$$

Denne er praktisk å bruke når vi skal diskretisere systemet.

3.10.1 Diskretiserer pitch-systemet

Fra (Haugen, 2003a)får vi

$$\Phi = e^{A*T_S} \tag{3.55}$$

Der T_S er sampelintervallet og vi kan bruker Matlab funksjonen *expm()*, og styrematrisen Δ får vi fra superposisjonsintegralet

$$\Delta = \int_0^{T_S} e^{A(T_S - x)} B \, dx \tag{3.56}$$

3.10.2 Den diskrete transferfunksjonen

Med dette kan vi finne den diskrete transferfunksjonen[4]

$$H(z) = \frac{y(z)}{u(z)} = C(zI - \Phi)^{-1}\Delta + D, \qquad \text{der } (zI - \Phi)^{-1} = \frac{\text{adj}(zI - \Phi)}{\det(zI - \Phi)} \tag{3.57}$$

Som gir[5]

$$H(z) = \frac{\begin{array}{c} -C_2\Delta_1(\Phi_{31}\Phi_{23} - \Phi_{21}\Phi_{33} + \Phi_{21}z) + \\ C_2\Delta_2(\Phi_{13}\Phi_{31} - \Phi_{11}\Phi_{33} + \Phi_{11}z + \Phi_{33}z - z^2) - \\ C_2\Delta_3(\Phi_{21}\Phi_{13} - \Phi_{11}\Phi_{23} + \Phi_{23}z) \end{array}}{\begin{array}{c} \Phi_{11}z^2 + \Phi_{22}z^2 + \Phi_{33}z^2 - z^3 + \Phi_{11}\Phi_{22}\Phi_{33} - \\ \Phi_{11}\Phi_{23}\Phi_{32} - \Phi_{12}\Phi_{21}\Phi_{33} + \Phi_{12}\Phi_{31}\Phi_{23} + \Phi_{21}\Phi_{13}\Phi_{32} \\ -\Phi_{13}\Phi_{22}\Phi_{31} - \Phi_{11}\Phi_{22}z + \Phi_{12}\Phi_{21}z - \Phi_{11}\Phi_{33}z + \Phi_{13}\Phi_{31}z \\ -\Phi_{22}\Phi_{33}z + \Phi_{23}\Phi_{32}z \end{array}} \tag{3.58}$$

[4] Bevis på hvordan vi kommer frem til diskret transferfunksjon

[5] Se vedlegg adj og det.

4 Utvikling av indre og ytre regulatorer

For å styre flyet bruker vi tre indre sløyfer og to ytre sløyfer. Indre sløyfe består i å stabilisere flyet i roll, pitch og yaw. Dette får vi til ved å ha en tilbakekobling fra vinkelhastighetene og stilingen til flyet.

Ytre sløyfe skal regulere posisjonen til flyet i forhold til jordkoordinater. Her bruker vi to vanlige PID-regulatorer som vi justerer inn ved hjelp av Ziegler- Nichols lukkede sløyfe metode.

4.1 Indre sløyfe

Polplasseringsmetoden blir brukt på de tre indre sløyfene til flyet.
Som nevnt tidligere skal vi linearisere likningene for vinkelhastighet og stilingen til flyet. Da er det enklest å bruke Eulervinkelrepresentasjonen, siden vi ønsker tre uavhengige indre sløyfer. Linearisererer[6] vi vinkelhastigheten og stilingen til flyet tre ganger med tanke på roll, pitch og yaw ut i fra følgende system

$$
\begin{aligned}
\dot{P} &= \frac{1}{I_{xx}I_{zz} - I_{xz}^2}\left(I_{zz}\bar{L} + I_{xz}N + \left(I_{yy}I_{zz} - I_{zz}^2 - I_{xz}^2\right)QR + I_{xz}\left(I_{xx} - I_{yy} + I_{zz}\right)PQ\right) \\
\dot{Q} &= \frac{1}{I_{yy}}\left(M + I_{xz}(R^2 - P^2) + (I_{zz} - I_{xx})PR\right) \\
\dot{R} &= \frac{1}{I_{xx}I_{zz} - I_{xz}^2}\left(I_{xz}\bar{L} + I_{xx}N + \left(I_{xz}^2 - I_{xx}I_{yy} + I_{xx}^2\right)PQ + I_{xz}\left(I_{yy} - I_{zz} - I_{xx}\right)QR\right) \\
\dot{\phi} &= P + \tan\theta\,(Q * \sin\phi + R * \cos\phi) \\
\dot{\theta} &= Q * \cos\phi - R * \sin\phi \\
\dot{\psi} &= \frac{Q * \sin\phi + R * \cos\phi}{\cos\theta}
\end{aligned}
\tag{4.1}
$$

Får man det lineariserte systemet i roll

$$
A_r = \begin{bmatrix} -\dfrac{I_{xz}Q\left(I_{xx} - I_{yy} + I_{zz}\right) + \frac{1}{2}V_T b^2 \rho s\left(C_{lP}I_{zz} + C_{nP}I_{xz}\right)}{I_{xz}^2 - I_{xx}I_{zz}} & 0 \\ 1 & \tan(\theta)(Q\cos(\phi) - R\sin(\phi)) \end{bmatrix}
$$

$$
B_r = \begin{bmatrix} -\dfrac{\frac{C_{l\delta R}I_{zz}V_T^2 b\rho s}{2} + \frac{C_{n\delta R}I_{xz}V_T^2 b\rho s}{2}}{I_{xz}^2 - I_{xx}I_{zz}} \\ 0 \end{bmatrix} \tag{4.2}
$$

Der $x = [P, \phi]^T$ og $u = \delta R$

Det lineariserte systemet i pitch

[6] se vedlegg linearisering

$$A_p = \begin{bmatrix} \frac{C_{mQ} V_T c^2 \rho\, s}{2\, I_{yy}} & 0 \\ \cos(\phi) & 0 \end{bmatrix}, \qquad B_p = \begin{bmatrix} \frac{C_{m\delta P}\, V_T^2\, c\, \rho\, s}{2\, I_{yy}} \\ 0 \end{bmatrix} \tag{4.3}$$

Der $x = [Q, \theta]^T$ og $u = \delta P$

Det lineariserte systemet i yaw

$$A_y = \begin{bmatrix} -\frac{\frac{C_{lR}\, I_{xz}\, V_T\, b^2\, \rho\, s}{2} - I_{xz}\, Q\, (I_{xx} - I_{yy} + I_{zz}) + \frac{C_{nR} I_{xx} V_T\, b^2\, \rho\, s}{2}}{I_{xz}^2 - I_{xx}\, I_{zz}} & 0 \\ \frac{\cos(\phi)}{\cos(\theta)} & 0 \end{bmatrix}$$
$$B_y = \begin{bmatrix} -\frac{\frac{C_{l\delta Y}\, I_{xz}\, V_T^2\, b\, \rho\, s}{2} + \frac{C_{n\delta Y}\, I_{xx} V_T^2\, b\, \rho\, s}{2}}{I_{xz}^2 - I_{xx} I_{zz}} \\ 0 \end{bmatrix} \tag{4.4}$$

Der $x = [R, \psi]^T$ og $u = \delta Y$

Vi lineariserer med et abreidspunkt der flyet ikke spinner, Eulervinklene er null og vi har en total hastighet på 200 $\left[\frac{m}{s}\right]$.

$$A_r = \begin{bmatrix} -33.3319 & 0 \\ 1 & 0 \end{bmatrix}, \qquad B_r = \begin{bmatrix} 2064.7 \\ 0 \end{bmatrix} A_p = \begin{bmatrix} -1.8616 & 0 \\ 1 & 0 \end{bmatrix},$$
$$B_p = \begin{bmatrix} -92.6425 \\ 0 \end{bmatrix} A_y = \begin{bmatrix} -7.4868 & 0 \\ 1 & 0 \end{bmatrix}, \qquad B_y = \begin{bmatrix} 215.2106 \\ 0 \end{bmatrix} \tag{4.5}$$

Vi velger å ha med integralleddet i regulatoren, slik at vi får null i statisk avvik. Vi får da de utvidede systemene

$$A_{n\,u} = \begin{bmatrix} A_n & \underline{0} \\ [0 & -1] \quad 0 \end{bmatrix}$$
$$B_{n\,u} = \begin{bmatrix} B_n \\ 0 \end{bmatrix} \tag{4.6}$$

Der n er de tre systemene for roll, pitch og yaw.

Vi ser om systemene er styrbare

$$\text{rang}(S) = \text{rang}([B \quad AB \quad A^2B \quad \dots \quad A^nB]) = n$$

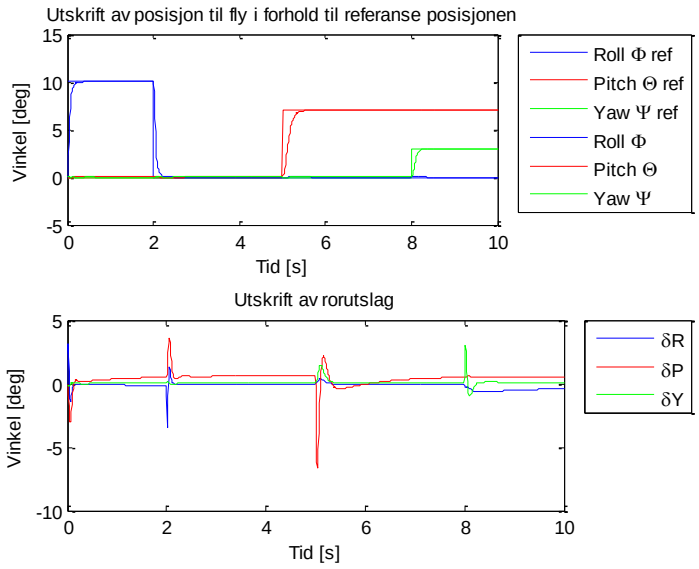
Siden de er styrbare er alle betingelser for polplassering med polynommetoden oppfylt og vi kan bruke metoden.

Med sammenfallende poler der knekkfrekvensen er $\omega_0 = 50$ for roll, $\omega_0 = 20$ pitch, og $\omega_0 = 30$ for yaw, får vi regulatorforsterkningen med Matlab funksjonen *place(Anu, Bnu, regpol)*

Tilbakekoblings forsterkninger	Integrator forsterkninger
$G_r = [0.0565 \quad 3.6349]$	$G_{rn} = -60.6021$
$G_p = [-0.6278 \quad -12.9629]$	$G_{pn} = -125.5848$
$G_y = [0.3835 \quad 12.5543]$	$G_{yn} = -125.5848$

Tabell 2: Regulator parametere med sammenfallende poler

Plotter ut og ser at vi får en nøyaktig og rask sporfølging av referansesignalet.



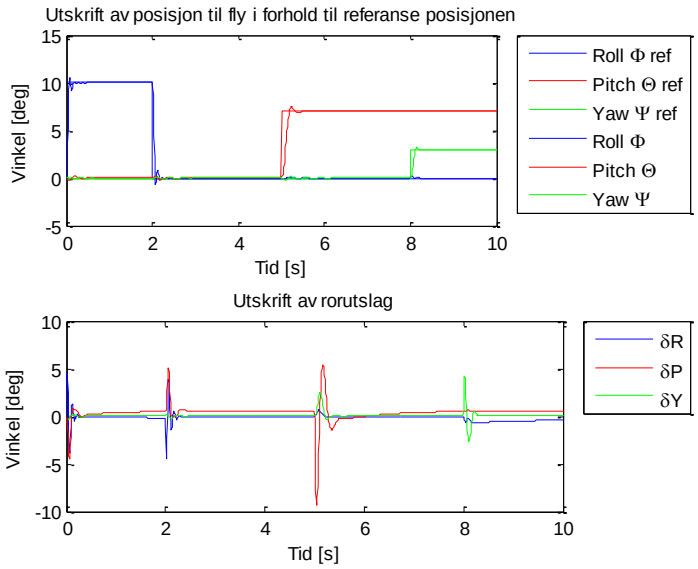
Figur 9: Vinkelforfølging med sammenfallendepoler

Med Butterworthpolynom der knekkfrekvensen er $\omega_0 = 50$ for roll, $\omega_0 = 20$ pitch, og $\omega_0 = 30$ for yaw, får vi regulatorforsterkningen med Matlab funksjonen *place(Anu, Bnu, regpol)*

Tilbakekoblings forsterkninger	Integrator forsterkninger
$G_r = [0.0323 \quad 2.4217]$	$G_{rn} = -60.5425$
$G_p = [-0.4117 \quad -8.6353]$	$G_{pn} = 86.3534$
$G_y = [0.2440 \quad 8.3639]$	$G_{yn} = -125.4585$

Tabell 3: Regulatorparametere med Butterworthpolynom

Plotter ut og ser at vi får en nøyaktig og rask sporfølging av referansesignalet.



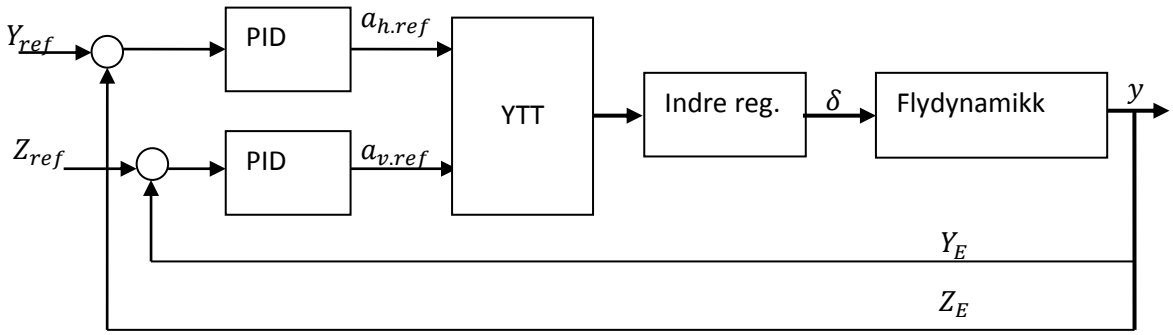
Figur 10: Vinkelforfølging med Butterworthpolynom

4.1.1 Konklusjon indre sløyfe

Vi kan se at vi får et litt mer urolig rorutslag med Butterworthpolynom og derfor blir det en litt mer urolig referanse forfølging. Siden begge metodene er nesten like raske er det anbefalt å bruke sammenfallende poler på indre sløyfe.

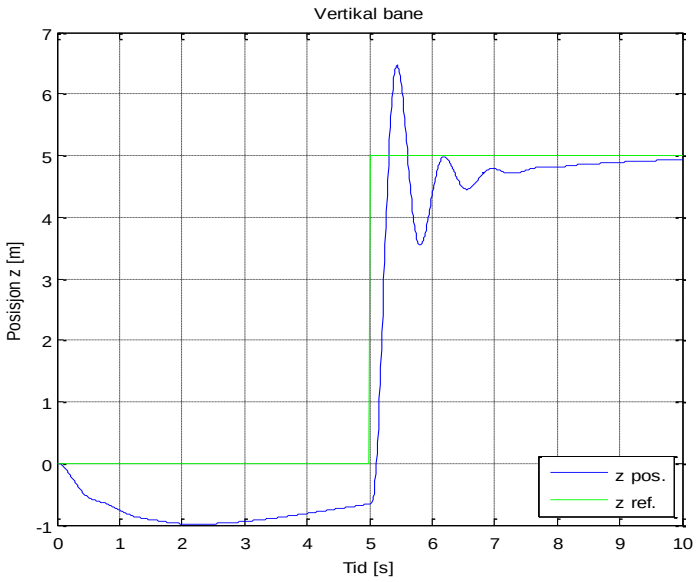
4.2 Ytre sløyfer med Yaw-To-Turn

Vi bruker Ziegler- Nichols lukkede sløyfe metode til å justere inn de to PID- regulatorene. Som vist på figuren under bruker vi en PID- regulator for å få en tilbakekobling fra posisjonene Y_E, Z_E, slik at man kan følge en gitt referanse. Her bruker vi ikke rull, bare pitch og yaw. Denne styremekanismen heter "Yaw To Turn"(YTT).



Figur 11: Ytre sløyfe med YTT

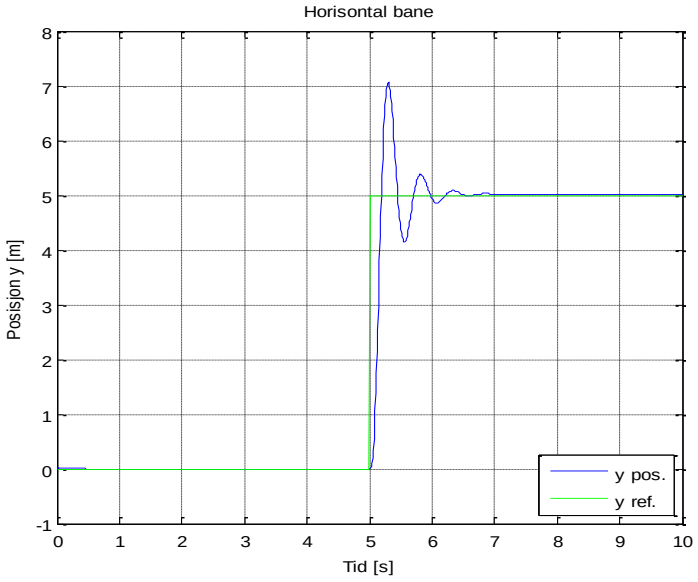
Vi bruker Ziegler- Nichols lukkede sløyfe på den ulineære modellen for å justere inn den vertikale regulatoren og finner dermed den kritiske forsterkningen $K_{pk.ver} = 0.047$ med den kritiske periodetiden $T_{p.ver} = 0.55/60$. Vi velger å bruke en PI-regulator da D-leddet vil føre til urolige rorutslag.



Figur 12: Plot av vertikal bane med Ziegler- Nichols lukkede sløyfe innstiling av PI- regulator

Her ser man at gravitasjonen fører til en forstyrrelse i signalet. Dette kan vi komme rundt ved å stramme inn regulatoren eller velge en annen regulator. Siden avviket er relativt lite velger vi å gå videre med denne regulatoren.

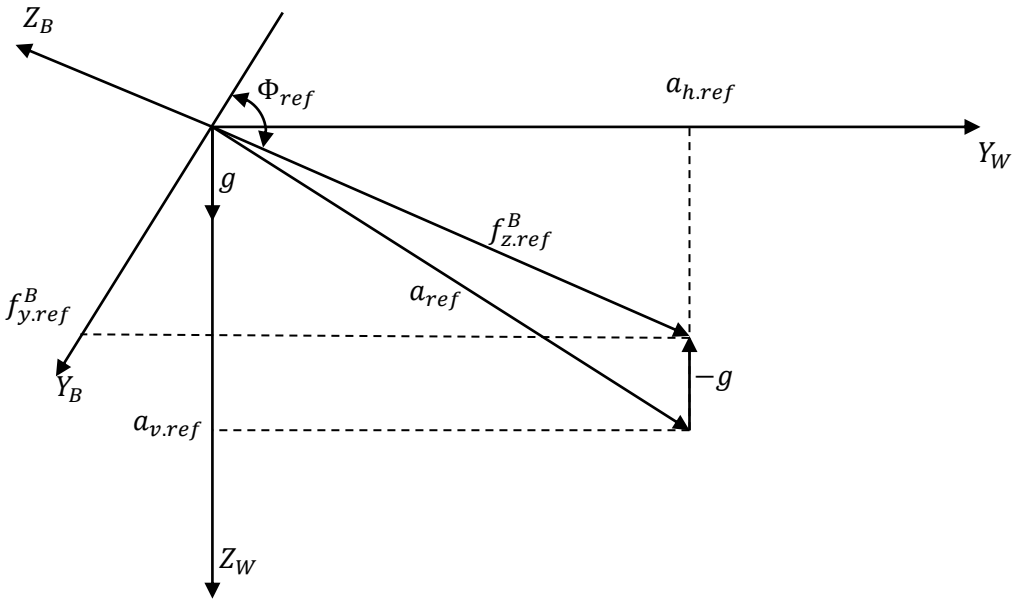
Deretter justerer vi inn den horisontale regulatoren med den vertikale regulatoren på, for å få best mulig sluttresultat. Da får vi den kritiske forsterkningen $K_{pk.hor} = 0.067$ som gir den kritiske perioden $T_{pk.hor} = 0.36/60$. Vi velger også her en PI- regulator, for at ikke rorutslaget skal bli for urolig.



Figur 13: Plot av horisontal bane med Ziegler- Nichols lukkede sløyfe innstiling av PI- regulator

Her ser vi at vi får svært bra følging av referansesignalet.

4.3 Innføring av Bank-To-Turn



Figur 14: Vektorrepresentasjon av BTT- prinsippet. Der Φ_{ref} er totale vinkelen mellom flykoordinatene og vindkoordinatene, g er gravitasjonen, a_{ref} er referanseakselerasjonen og f_{ref} er akselerasjon sett fra flyet.

I styresystemet til flyet bruker vi Bank– To– Turn (BTT) prinsippet. Som navnet tilsier så ruller flyet for å svinge, på denne måten utnyttes flyets aerodynamikk på best mulig måte. Dette får man til ved å se på akselerasjonen i vertikal- og horisontal- planet som vist på figuren over.

Ut i fra figuren over får man

$$\begin{aligned} \Phi_{ref} &= \operatorname{atan2}\left(a_{h.ref}, g - a_{v.ref}\right) f^B_{z.ref} = K_1\left(-a_{h.ref}\sin(\phi) + a_{v.ret}\sin(\phi)\right) \\ a'_{v.ref} &= g - a_{v.ref} \\ f^B_{y.ref} &= K_1\left(a_{h.ref}\cos(\phi) + a'_{v.ref}\sin(\phi)\right) \end{aligned} \tag{4.7}$$

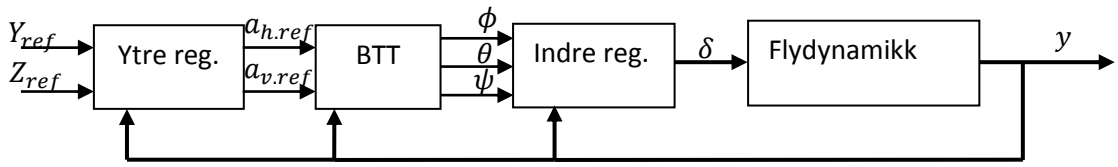
Her er ϕ rullvinkelen til flyet i pitch.

Siden vi skal ha BTT referansen i rull, pitch og yaw kan man multiplisere med en faktor K_{btt1}, K_{btt2} og K_{btt3}.

$$\begin{aligned} \phi_{ref} &= K_{bbt1}\operatorname{atan2}\left(a_{h.ref}, g - a_{v.ref}\right) \\ \theta_{\mathrm{ref}} &= K_{bbt2}\left(-a_{h.ref}\sin(\phi) + a_{v.ret}\sin(\phi)\right) \\ \psi_{ref} &= K_{bbt3}\left(a_{h.ref}\cos(\phi) + a'_{v.ref}\sin(\phi)\right) \end{aligned} \tag{4.8}$$

Hvis vi skal ta ekstremt krappe svinger må vi justere ned K_{bbt1}og juster opp K_{btt3} slik at vi får mindre rull- i forhold til yaw- vinkel. Med dette vil vi unngå at flyet blir ustabilt i krappe svinger. Noe som blir diskutert videre i kapittel 5.

Vist i blokkdiagram får vi følgende system



Figur 15: Blokkdiagram av flymodellen med BTT

4.3.1 Tester ut Bank-To-Turn prinsippet på den ulineære modellen

Etter testing har vi funnet to stilinger på BTT der $K_{btt1} = 5, K_{btt2} = 1$ og $K_{btt3} = 0{,}1$ gir et lite overlegg/rull, mens $K_{btt1} = 10, K_{btt2} = 1$ og $K_{btt3} = 0{,}05$ gir ett stort overlegg.

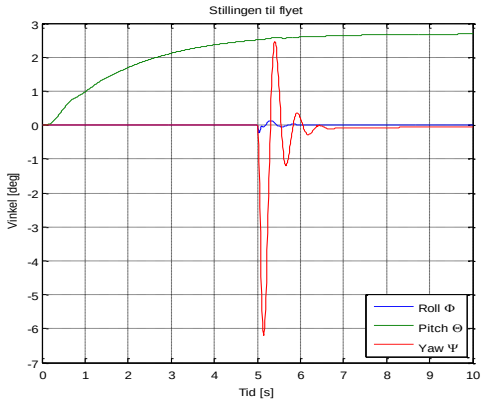


Figure 16: Eulervinklene med YTT

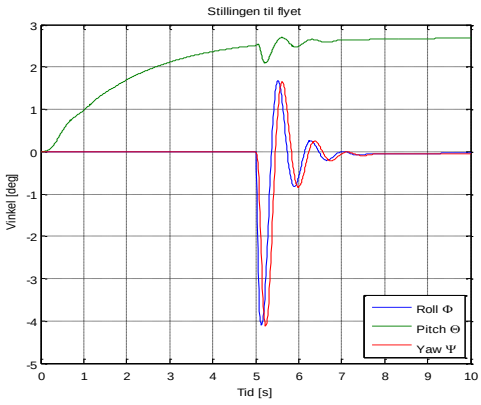
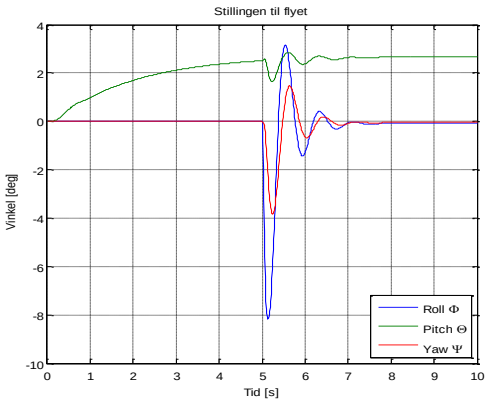


Figure 17: Eulervinklene med liten BTT



Figur 18: Eulervinklene med stor BTT

Noe vi må være oppmerksomme på er at stabiliteten til flyet blir dårligere jo mer man legger over flyet for å svinge

4.4 Setter opp referansebane

Det er et ønske å styre flyet etter referansepunkter i rommet, dette kan vi gjøre ved hjelp av spline eller interpolasjon polynom metoder som (Fossen, 2002 ss. 152-169). Vi velger å bruke Hermite

interpolasjons polynom som trekker en bane mellom referansepunktene ut fra den første deriverte og endepunktene er valgt slik at de tilfredsstiller start og sluttbetingelsene. Siden spline bruker den andre deriverte får man en noe mer svingete bane og er dermed ikke ønskelig å bruke.

Vi bruker Matlab funksjonen *pchip()* som gir oss banen i vertikal- og horisontal- planet ut i fra Hermite interpolasjon polynomet. Vi får banen i rommet som vist under, figur 23 og 24.

4.5 Simulerer fullt system ut fra referansepunkter

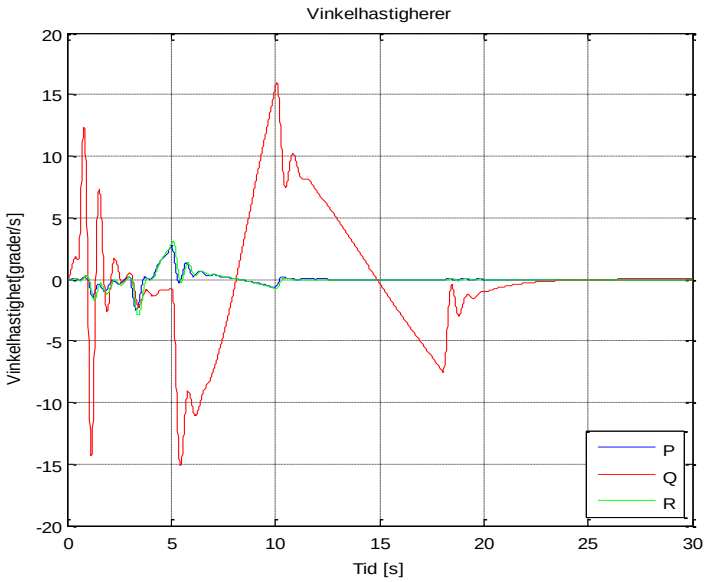
Vi velger referansepunktene ut i fra tabellen under.

Tid [s]	0	0,5	1	3	5	10	18	20
X- posisjon [m]	0	100	200	600	1000	2000	3600	4000
Y- posisjon [m]	0	0	0	10	30	0	−10	−10
z- posisjon [m]	0	0	5	5	−10	−350	20	20

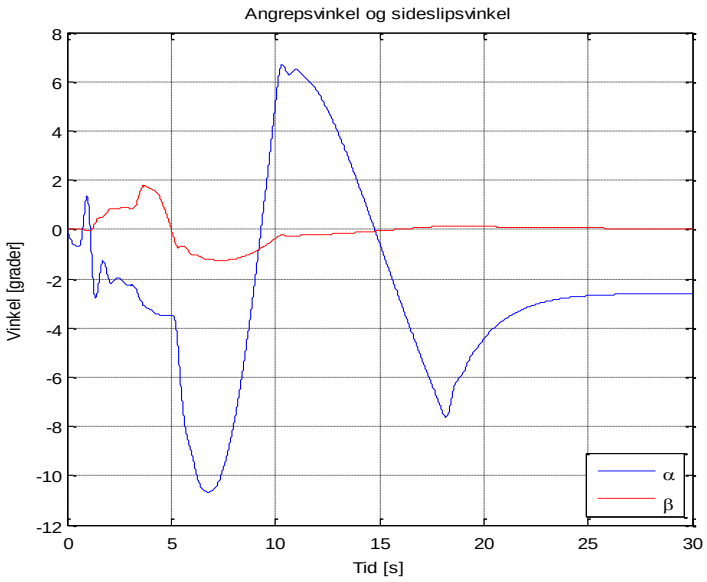
Tabell 4: Referanse punkter i rommet

Her har vi tatt med X- posisjonsavhengigheten, dette for at det lettere skal kunne implementere en hastighetskontroll som kan være avhengig av X- posisjonen.

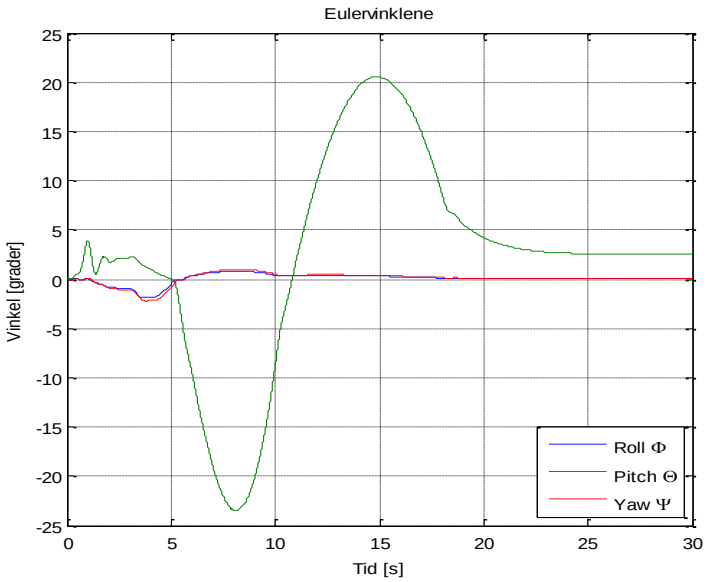
Med alt dette til grunde ser man at en får bra forfølging av referansesignalet selv om vi tar en svært krapp sving som gir en angrepsvinkel på ca. 11^0.



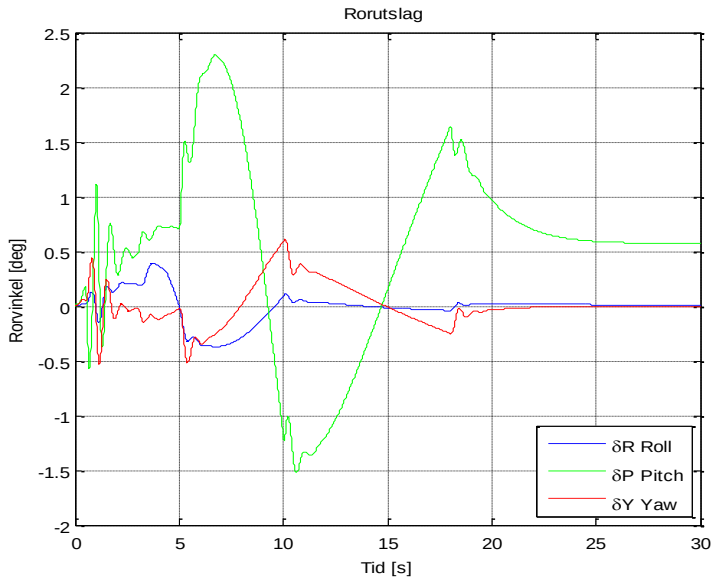
Figur 19: Vinkelhastighetene i roll, pitch og yaw



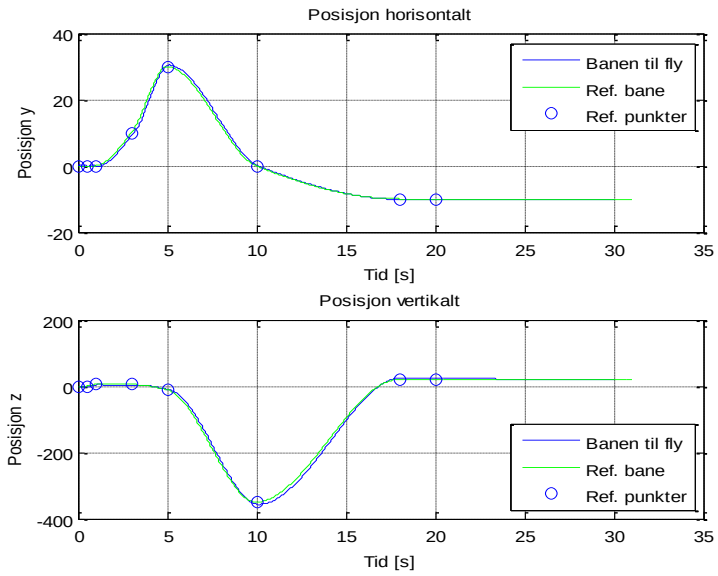
Figur 20: Angrepsvinkelen og sideslipsvinkelen



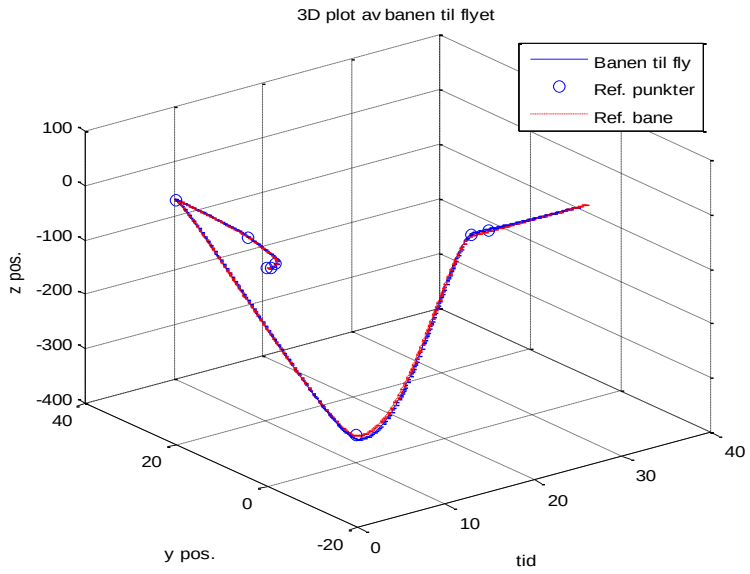
Figur 21: Eulervinklene til flymodellen



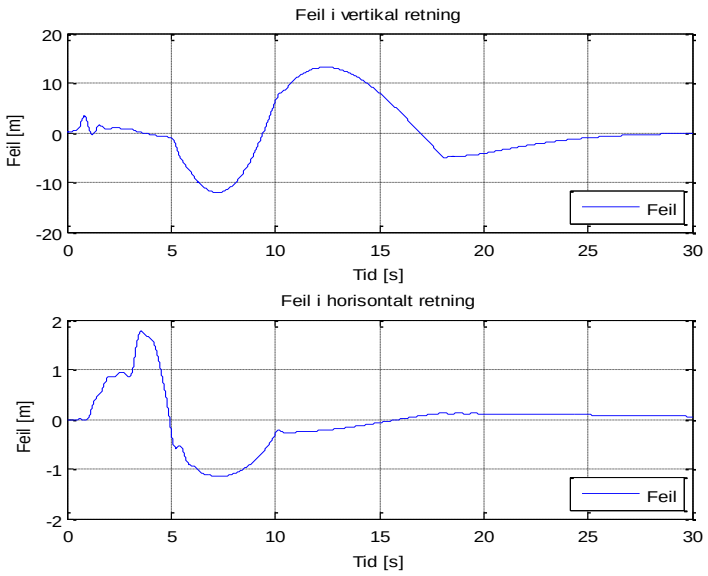
Figur 22: Rorutslag



Figur 23: 2D plot av banen til flymodellen i forhold til referansebane og refeansepunkter



Figur 24: 3D plot av banen til flymodellen i forhold til referansebane og refeansepunkter



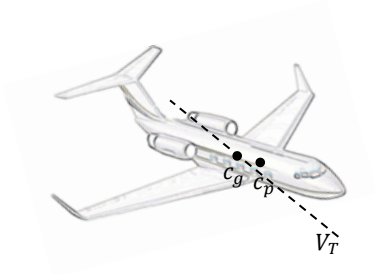
Figur 25: Feil fra referansebanen i vertikal og horisontal retning

5 Stabilitet til flyet

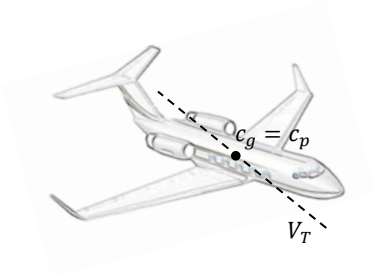
I dette kapitelet skal vi se på stabiliteten til flyet med kommersiell regulator der vi har konstante aerodynamiske koeffisienter. Deretter skal vi se på stabiliteten til flyet når vi har forandring i den aerodynamiske momentkoeffisienten C_M.

5.1 Stabiliteten til flyet med konstante aerodynamiske koeffisienter

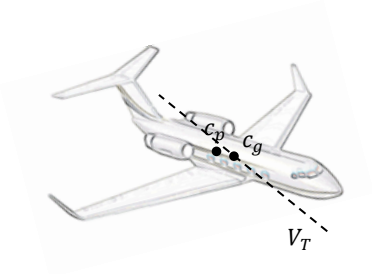
Flyets stabilitet er avhengig av hvor tyngdepunktet i flyet ligger. Hvis tyngdepunktet c_g ligger foran senter av total hastighet c_p vil flyet oppføre seg som en kastepil/ dart som er et stabilt flygende legeme. Hvis vi derimot kaster kastepilen/ darten med nesen feil vei vil den snu seg i luften. Tyngdepunktet ligger bak senter av total hastighet og vi har et ustabilt flygende legeme. Hvis $c_g = c_p$ har vi et marginalt stabilt flygende legeme.



Figur 26: Tyngdepunktet vs. sentrum av total hastighet, her har vi et ustabilt fly



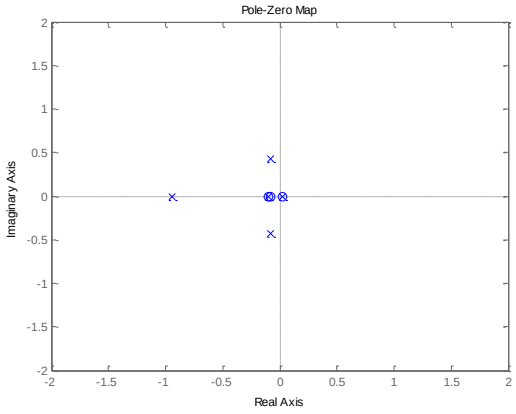
Figur 27: Tyngdepunktet vs. sentrum av total hastighet, her har vi et marginalt stabilt fly



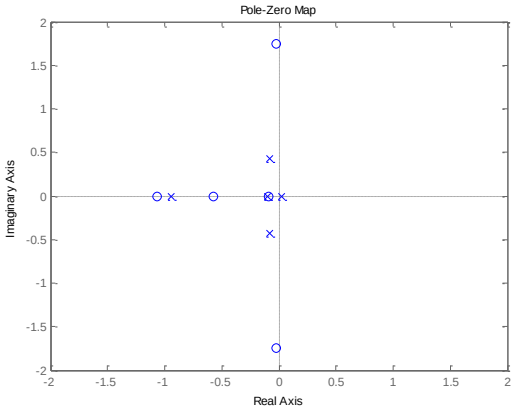
Figur 28: Tyngdepunktet vs. sentrum av total hastighet, her har vi et stabilt fly

Som vi skal se i delkapittel 5.2 er det $C_{M\alpha}$ som har mest å si på stabiliteten til flyet. Når denne er negativ har vi et stabilt system, hvis vi ser på det forenklede systemet for pitch. Mens det er et ustabilt system når den er positiv. Dette kan vi se ut i fra polplasseringen til det forenklede systemet. Hvis vi tar med alle avhengighetene til pitch ser man at det er poler i venstre halvplan for det kontinuerlige systemet. Dette viser at man har et ustabilt system.

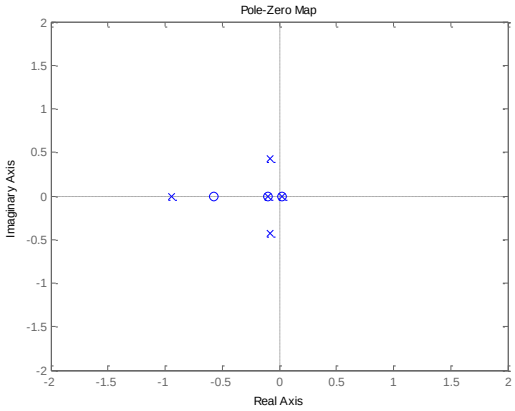
Som vist på pol-plottene for Eulervinklene roll, pitch og yaw ser man at de er ustabile systemer.



Figur 29: Polplott for åpen sløyfe for kontinuerlig system i roll

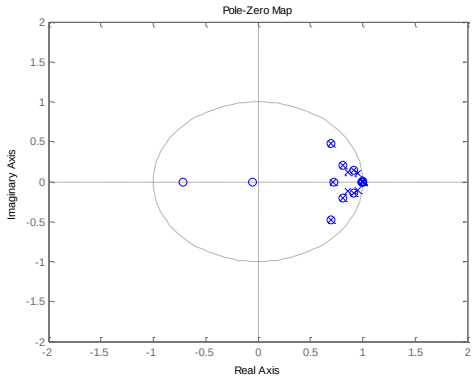


Figur 30: Polplott for åpen sløyfe for kontinuerlig system i pitch

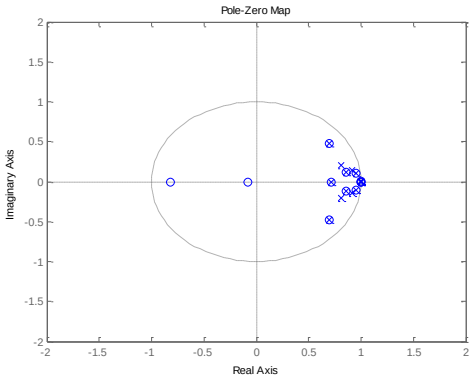


Figur 31: Polplott for åpen sløyfe for kontinuerlig system i yaw

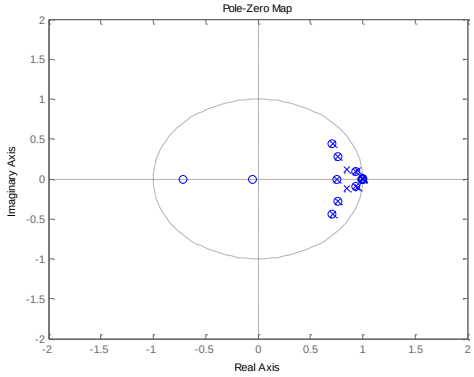
Ser man på hvilken effekt BTT har på stabiliteten til flymodellen ser man på plottene under at de horisontale polene beveger seg utover mot høyre jo mer man legger flyet over for å svinge. Derav ser man at systemet blir mer ustabilt jo mer vi legger over flyet for å svinge.



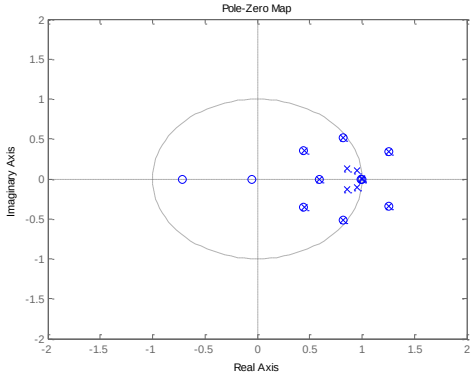
Figur 32: Polplott av lukket sløyfe med YTT for diskret vertikal sløyfe



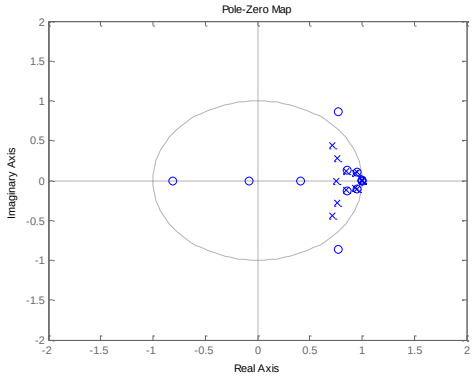
Figur 33: Polplott av lukket sløyfe med YTT for diskret horisontal sløyfe



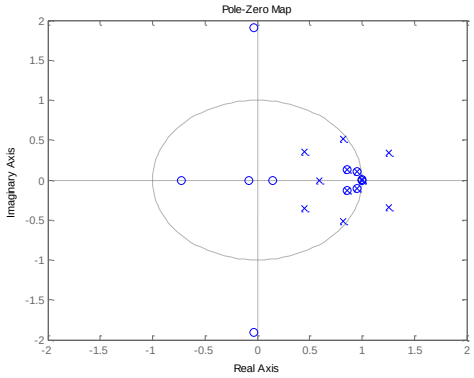
Figur 34: Polplott av lukket sløyfe med liten BTT for diskret vertikal sløyfe



Figur 35: Polplott av lukket sløyfe med stor BTT for diskret vertikal sløyfe



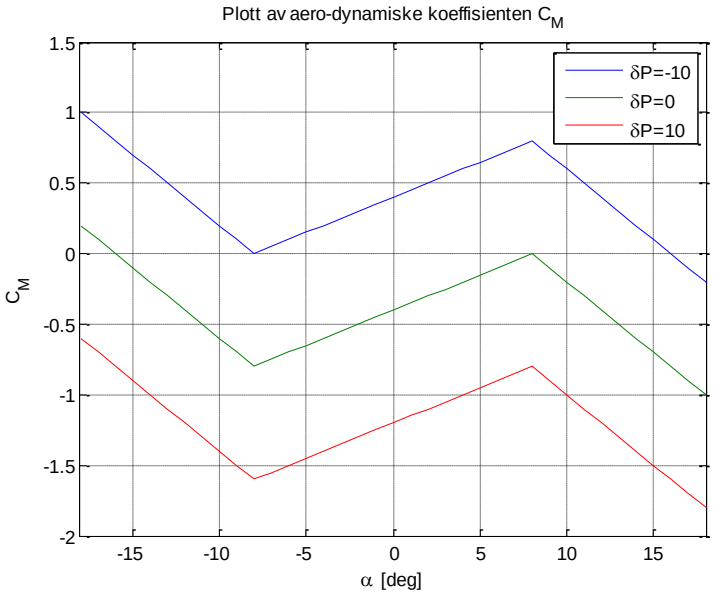
Figur 36: Polplott av lukket sløyfe med liten BTT for diskret horisontal sløyfe



Figur 37: Polplott av lukket sløyfe med stor BTT for diskret horisontal sløyfe

5.2 Stabiliteten til flyet med variable aerodynamiske koeffisienter

Flyets stabilitet varierer med angrepsvinkelen. Hovedgrunnen til dette er at flyet har forskjellig areal mot vinden og dermed får vi en variasjon av aerodynamikken. Dette skal vi se på ved å variere koeffisienten C_M, som er koeffisienten for moment i pitch.



Figur 38: Plott av momentkoeffisienten for pitch

Som vi kan se ut i fra figuren over har vi en skarp knekk, noe som er svært relevant i aerodynamikken. Grunnen til dette er at man kan få turbulens rundt vingen som fører til store forandringer i stabiliteten til flyet.

Her kan vi lese ut koeffisienten C_{M0} som er

$$C_{M0} = -0.4 \tag{5.1}$$

Hvis vi partiellderiverer C_M med tanke på angrepsvinkelen α får vi $C_{M\alpha}$

$$C_{M\alpha}|_{-8\leq\alpha\leq 8} = \frac{C_M(\alpha_{maks}) - C_M(\alpha_{min})}{\alpha_{maks} - \alpha_{min}} = \frac{0-(-0.4)}{8-0} = 0.05\left[{}^{1}/_{deg}\right]$$
$$C_{M\alpha}|_{8<\alpha<\infty\ \&\ -\infty<\alpha<-8} = \frac{-1.0-0}{18-8} = -0.1\left[{}^{1}/_{deg}\right] \tag{5.2}$$

Ser at avstanden mellom δP er lik og vi får en konstant $C_{M\delta P}$ som vi finner med partiellderivasjon med tanke på pådraget δP

$$C_{M\delta P} = \frac{C_M(\delta P_{-10}, \alpha) - C_M(\delta P_0, \alpha)}{\delta P_{-10} - \delta P_0} = \frac{0.4-(-0.4)}{-10-0} = -0.08\left[{}^{1}/_{deg}\right] \tag{5.3}$$

Der vi kan gjøre om $C_{M\delta P}$ til å gjelde for våre fire styreflater

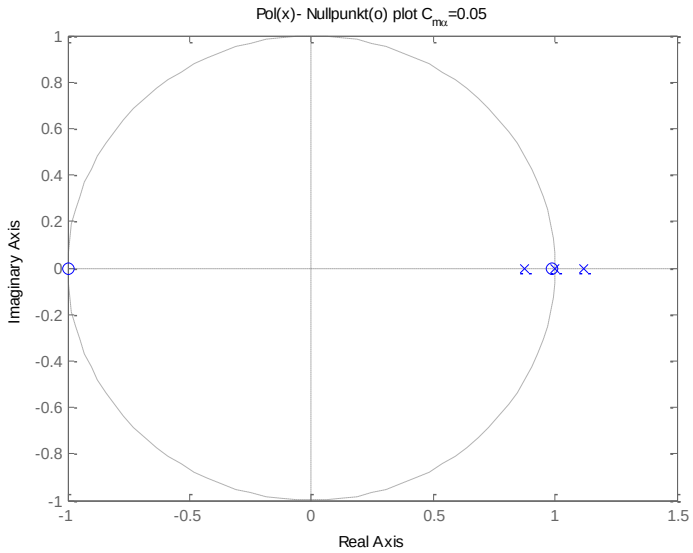
$$\delta P = \delta c_1 + \delta c_2 + \delta c_3 + \delta c_4 \tag{5.4}$$

Siden styreflaten er like får vi

$$C_{M\delta c_n} = \frac{C_{m\delta P}}{4} = -0.02 \left[{}^{1}/_{deg}\right] \tag{5.5}$$

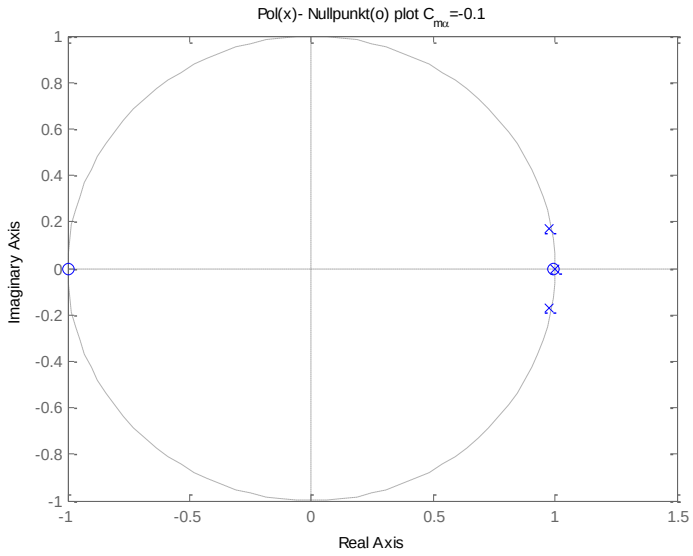
Der n er styreflate nummer 1-4.

Med dette til grunne kan vi se på stabiliteten til flyet med hjelp av polplasseringen.



Figur 39: Polplott av den diskrete pitch-system der $C_{M\alpha} = 0.05 \left[{}^{1}/_{deg}\right]$

Her ser vi at vi har en diskret pol utenfor enhetssirkelen som gjør systemet ustabilt.



Figur 40: Polplott av den diskrete pitch-system der $C_{M\alpha} = -0.1\ \left[{}^{1}/_{deg}\right]$

Her har vi et marginalt stabilt system siden vi har en pol på enhetssirkelen. Her ser vi at $C_{M\alpha}$-avhengigheten får systemet mer stabilt. Lineariserer vi det fulle systemet vil vi se at vi har ustabile modier også når C_M har negativ kurve.

5.2.1 Konklusjon

Når systemet går fra $C_{M\alpha} = 0.05$ til $C_{M\alpha} = -0.1$ vil vi få en tregere respons i systemet hvis vi ikke forandrer regulatoren når denne overgangen forekommer. Derfor er det ønskelig å bruke en adaptiv regulator på systemet slik at vi kan få raskest mulig respons under alle forhold. Dette skal vi se på i de neste kapitlene.

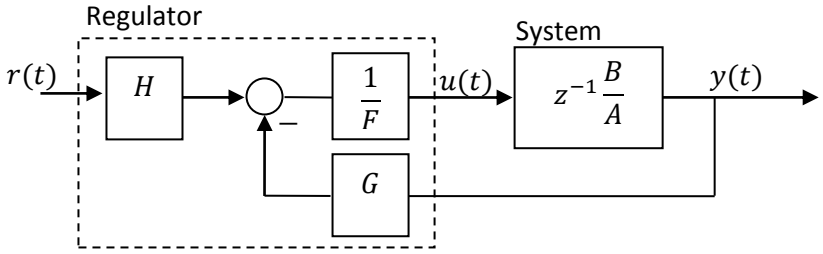
6 Regulerer pitch-systemet med forskjellige $C_{M\alpha}$

I dette kapittelet skal vi komme frem til potildelingsmetoden. Der vi har brukt (Wellstead, et al., 1991 ss. 243-308) som støttelitteratur. Deretter skal vi bruke det forenklede pitch-systemet for å verifisere regulatoren med forskjellige $C_{M\alpha}$.

6.1 Poltildelings metoden

Ta utgangspunkt i en ARMAX modell som i Adaptive miljøer kalles CARMA (Controlled Auto-Reressive Moving Average)

$$Ay(k) = Bu(k) + Ce(k); \tag{6.1}$$



Figur 41:Regulatorstruktur for poltildelings metoden

Tar utgangspunkt i regulatorstrukturen i figuren over

$$Fu(k) = Hr(k) - Gy(k) \tag{6.2}$$

$$(FA + BGz^{-1})y(k) = z^{-1}BHr(k) + CFe(k) \tag{6.3}$$

Ved å kombinere regulator og prosess får vi lukket sløyfe.

$$\frac{y}{r}(k) = \frac{\left(\frac{HBz^{-1}}{FA}\right)}{1 + \frac{GBz^{-1}}{FA}} = \frac{HBz^{-1}}{FA + BGz^{-1}} \tag{6.4}$$

$$\frac{y}{e}(k) = \frac{\frac{C}{A}}{1 + \left(\frac{GBz^{-1}}{FA}\right)} = \frac{\frac{A}{C}}{\frac{FA + GBz^{-1}}{FA}} = \frac{CF}{FA + z^{-1}BG} \tag{6.5}$$

Polene i lukket sløyfe systemet kan velges ved å spesifisere T i polynomidentiteten (Diophntus-identiteten)

$$FA + z^{-1}BG = TC \tag{6.6}$$

Hvor F, G, H polynomene er gitt av:

$$F = 1 + f_1 z^{-1} + \cdots + f_{n_f} z^{-n_f} \tag{6.7}$$

$$G = g_0 + g_1 z^{-1} + \cdots + g_{n_g} z^{-n_g}$$
$$H = h_0 + h_1 z^{-1} + \cdots + h_{n_h} z^{-n_h}$$
$$\boxed{T = 1 + t_1 z^{-1} + \cdots + t_{n_t} z^{-n_t}} \rightarrow polene\ i\ lukket\ sløyfe$$

For en entydig løsning av (6.6)(3.49) må

$$\begin{aligned} n_f &= n_b \\ n_g &= n_a - 1 \end{aligned} \tag{6.8}$$

Forutsatt at A og B ikke har felles nullpunkter. Dette betyr at systemet skal være styrbart og observerbart.

$$n_t \leq n_a + n_b - n_c \tag{6.9}$$

Ved å sette (6.6) inn i systemlikningen (6.3) får vi

$$y(k) = \frac{HB}{TC} r(k-1) + \frac{F}{T} e(k) \tag{6.10}$$

Vi antar at C polynomet er kansellert i støyleddet, dette krever at C^{-1} er stabil, en antagelse som er svak.

H, den såkalte prekompensatoren velges slik at man får tilnærmet eksakt forfølging ved langsom varierende referanse og kanselering av C i støyleddet.
Det enkleste valget blir:

$$H = C \left(\frac{T}{B}\right)_{z=1} \tag{6.11}$$

og

$$y(k) = \left(\frac{T}{B}\right)_{z=1} \left(\frac{B}{T}\right) r(k-1) + \frac{F}{T} e(k) \tag{6.12}$$

Får transferfunksjonen:

$$y(k) = 1 * r(k-1) \tag{6.13}$$

Felles faktor i B og T polynomet svarer til ikke observerbare modi i det lukkede systemet. Disse kan påvirke utseende på utgangssignalet mellom sampel-tidspunktene og bør ikke ligge for langt fra origo.
Felles faktorer i H og T polynomet svarer til ikke styrbare modi i det lukkede systemet. (Dette er lite heldig, spesielt for lave frekvenser)

For å løse ut F, G, H ser vi på polynomlikningen

$$FA + z^{-1} BG = T \tag{6.14}$$

for $n_a = 3, n_b = 2$ får vi

$$(Hf_1z^{-1} + f_2z^{-2})(1 + a_1z^{-1} + a_2z^{-2} + a_3z^{-3}) + z^{-1}(b_0 + b_1z^{-1} + b_2z^{-2})(g_0 + g_1z^{-1} + g_2z^{-2} = 1 + t_1z^{-1} \quad (6.15)$$

Løser man ut F og G, ved å samle leddene

$$\begin{aligned} &f_1 + b_0g_0 = t_1 - a_1 \\ &a_1f_1 + f_2 + b_1g_0 + b_0g_1 = t_2 - a_2 \\ &a_2f_1 + a_1f_2 + b_2g_0 + b_1g_1 + b_0g_2 = t_3 - a_3 \\ &a_3f_1 + a_2f_2 + b_2g_1 + b_1g_2 = t_4 \\ &a_3f_2 + b_2g_2 = t_5, \end{aligned} \quad (6.16)$$

Som vi kan skrive på matriseform

$$\begin{bmatrix} 1 & 0 & b_0 & 0 & 0 \\ a_1 & 1 & b_1 & b_0 & 0 \\ a_2 & a_1 & b_2 & b_1 & b_0 \\ a_3 & a_2 & 0 & b_2 & b_1 \\ 0 & a_3 & 0 & 0 & b_2 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} t_1 - a_1 \\ t_2 - a_2 \\ t_3 - a_3 \\ t_4 \\ t_5 \end{bmatrix} \quad (6.17)$$

$$A_r\theta_r = b_r \quad (6.18)$$

Ut fra likning (6.17) får vi den generelle formen

$$\begin{bmatrix} 1 & 0 & \dots & b_0 & 0 & \dots & 0 \\ a_1 & \ddots & 0 & b_1 & b_0 & \ddots & \vdots \\ \vdots & \ddots & 1 & \vdots & \ddots & \ddots & 0 \\ a_{n_a} & \ddots & a_1 & b_{n_b} & \dots & \dots & b_0 \\ 0 & \ddots & \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \dots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & a_{n_a} & 0 & \dots & 0 & b_{n_b} \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_{nf} \\ g_0 \\ \vdots \\ g_{ng} \end{bmatrix} = \begin{bmatrix} t_1 - a_1 \\ \vdots \\ t_{n_a} - a_{n_a} \\ t_{n_a+1} \\ \vdots \\ t_{n_t} \end{bmatrix} \quad (6.19)$$

Forutsatt at A_r og B_r ikke har felles faktor (coprime), er av riktig dimensjon $(a_3 \neq 0, b_2 \neq 0)$ og ${A_r}^{-1}$ eksisterer, kan vi bruke standardmetoden for å finne parametervektoren θ_r

$$\theta_r = {A_r}^{-1}b_r \quad (6.20)$$

Eller med Matlab koden

$$\theta_r = A_r \backslash b_r \quad (6.21)$$

Etter å ha løst (6.20) blir det lukkede systemet av formen

$$y(k) = \frac{BH}{T}r(k-1) \quad (6.22)$$

For å sikre at systemet følger referansen kan vi velge
H =konstant ved DC verdi $(t \to \infty, z \to 1)$

$$H = \frac{T}{B}|_{z=1} \quad (6.23)$$

Dette valget betyr at det lukkede systemet får transfer funksjonen$= 1$ ved lav frekvens (DC verdi).
En annen metode som kan gi en bedre trensientrespons, er å velge fornuftig nullpunkter i H. (cut and tray).
Fra (6.22) ser vi at BH vil påvirke transientresponsen. En løsning er å velge

$$H = \frac{1}{B} \tag{6.24}$$

B polynomet kan være invers ustabilt og en kanselering vil bli lite nøyaktig.(Er ikke anbefalt, hvis vi ikke kjenner prosessen eksakt.)

Med dette til grunne får vi følgende algoritme:

1. Setter opp ønskelige poler

$$T = (s - p_1)(\dots)\left(s - p_{n_t}\right) \tag{6.25}$$

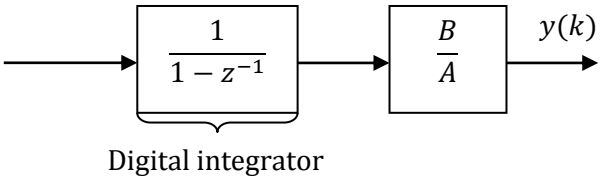
2. Setter opp A_r matrisen ut i fra A og B
3. Finner θ_r
4. Deler opp θ_r

$$\begin{aligned} \theta_r(1{:}\,n_f) &= F \\ \theta_r\left((n_f+1){:}\,length(\theta_r)\right) &= G \end{aligned} \tag{6.26}$$

Denne algoritmen passer å bruke sammen med RLS og kontinuerlig diskretisering av pitch transfer funksjonen.

6.1.2 Inkrement kontroll

Ofte vil vi ha bruk for et integralledd i regulatoren – kalles inkrement kontroll



Figur 42: Inkrement kontroll

Denne type regulatorer kan effektivt benyttes på systemer av formen

$$Ay(k) = Bu(k-1) + Dv(k) + N(k) \tag{6.27}$$

hvor

$v(k)$ er en ukjent forstyrrelse
$N(k)$ kan være en drift eller konstant forstyrrelse

$$N(k) = n_0 + n_1 k + \cdots + n_n k^{nn} \tag{6.28}$$

NB! Drift er avhengig av tiden.
En konstant forstyrrelse kan representeres ved en verdi på n_0.

$$D = d_0 + d_1 z^{-1} + \cdots + d_{nd} z^{-nd} \tag{6.29}$$

Den totale forstyrrelsen som går inn i systemet kan skrives

$$S(k) = \frac{N(k)}{A} v(k) + \frac{C}{A} e(k) \tag{6.30}$$

(6.27) kan skrives på inkrement form

$$\bar{A} y(k) = Bu(k-1) + \bar{D} v(k) + \overline{N}(k) \tag{6.31}$$

Hvor

$$\bar{A} = (1 - z^{-1})A = \Delta A\bar{D} = (1 - z^{-1})A = \Delta D\overline{N} = (1 - z^{-1})A = \Delta N \tag{6.32}$$

Antar at kontrolleren er på formen

$$\Delta u(k) = -\frac{C}{F}\big(y(k) - r(k)\big) \tag{6.33}$$

G og F må velges for å tilfredsstille likning(6.32)

$$F\overline{A} + z^{-1}BG = T \tag{6.34}$$

Orden (grad) til G må økes med én for at (6.34) skal få en entydig løsning.
Lukket sløyfe respons blir

$$y(k) = \frac{GB}{T} r(k-1) + \frac{F\overline{D}}{T} v(k) + + \frac{F}{T}\overline{N}(k) \tag{6.35}$$

Som kan skrives på formen

$$y(k) = r(k) + \frac{1 - z^{-1}}{T}\big(FDv(k) - FAr(k) + FN(k)\big) \tag{6.36}$$

6.3 Finner passende poler for flyet med forskjellige $C_{M\alpha}$

Siden vi skal regulere flyet som har en varierende C_M må vi sjekke at poltidelingsregulatoren greier å følge en gitt referanse. Først tester vi regulatoren på det stabile systemet der $C_{M\alpha} = -0.1\left[\frac{1}{deg}\right]$, $C_{M0} = -0.4$ og $C_{M\delta P} = -0.02\left[\frac{1}{deg}\right]$ (C_{M0} er ubenevnt). Her ser vi på C_{M0} som en forstyrrelse. Vi bruker det forenklede systemet for Pitch som vi har funnet i kapittel 3. Ut i fra det finner vi transferfunksjonen for en angrepsvinkel α på 0 til 8 grader når $C_{m\alpha} = 0.05\left[\frac{1}{deg}\right]$ og $V_T = 200\left[\frac{m}{s}\right]$ får vi

$$H(z) = \frac{-0.01176\, z^2 - \ 0.0001224z + 0.01158}{z^3 - \ 2.994z^2 + \ 2.973z - \ 0.9793} \tag{6.37}$$

Med en angrepsvinkel α på 8 til 15 grader når $C_{m\alpha} = -0.10\left[\frac{1}{deg}\right]$ og $V_T = 200\left[\frac{m}{s}\right]$ får vi

$$H(z) = \frac{-0.01176z^2 - 5.256e-005z + 0.01155}{z^3 - 2.95z^2 + 2.929z - 0.9793} \quad (6.38)$$

Polene til det åpne systemet når $C_{M\alpha} = -0.1$

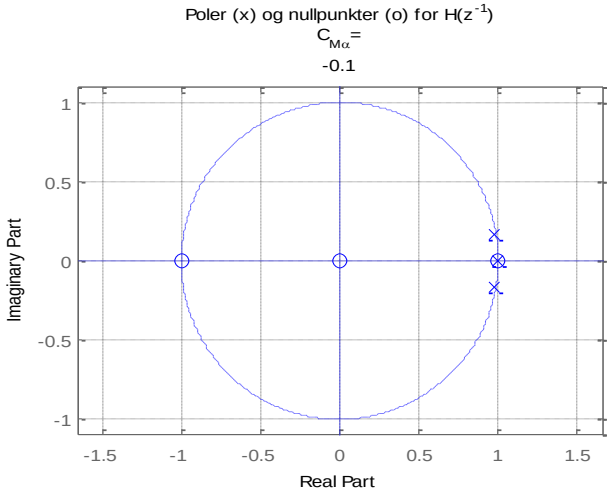
$$P_z = \begin{bmatrix} 1.0 \\ 0.9751 + 0.169i \\ 0.9751 - 0.169i \end{bmatrix} \quad (6.39)$$

Her ser vi at vi har et marginalt stabilt system.

Med litt 'cut and tray' får vi følgende sammenfallende poler for det lukkede-systemet

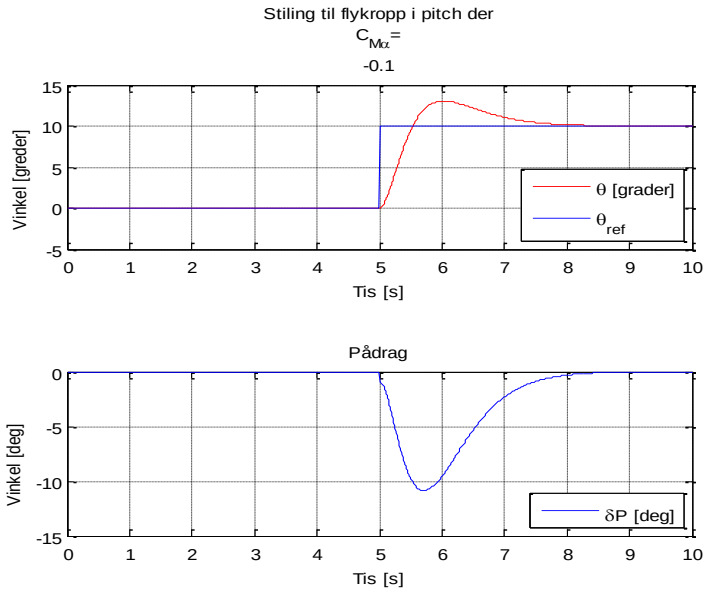
$$T_p = \begin{bmatrix} 0.9724 \\ 0.9724 \\ 0.9724 \end{bmatrix} \quad (6.40)$$

For sampel tiden på $T_s = 0.01\ s$



Figur 43: Poler og nullpunkter for pitchsystem med $C_{M\alpha} = -0.1\ \left[\frac{1}{deg}\right]$

Vi får en nøyaktig og rask forfølgelse av referansen



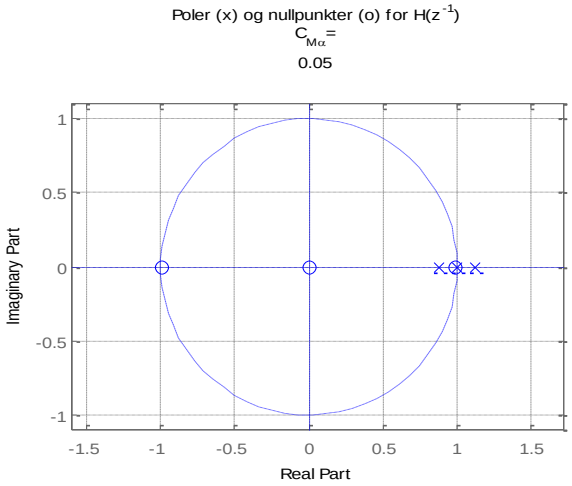
Figur 44: Følging av step i referansen for pitch-systemet med $C_{M\alpha} = -0.1\ \left[\frac{1}{deg}\right]$

Der pådraget er mulig å følge for en servomotor som skal styre roret.

Med et ustabilt system, der $C_{M\alpha} = 0.05\left[\frac{1}{deg}\right]$, $C_{M0} = -0.4$ og $C_{M\delta P} = -0.02\left[\frac{1}{deg}\right]$ har man da følgende åpen sløyfe poler

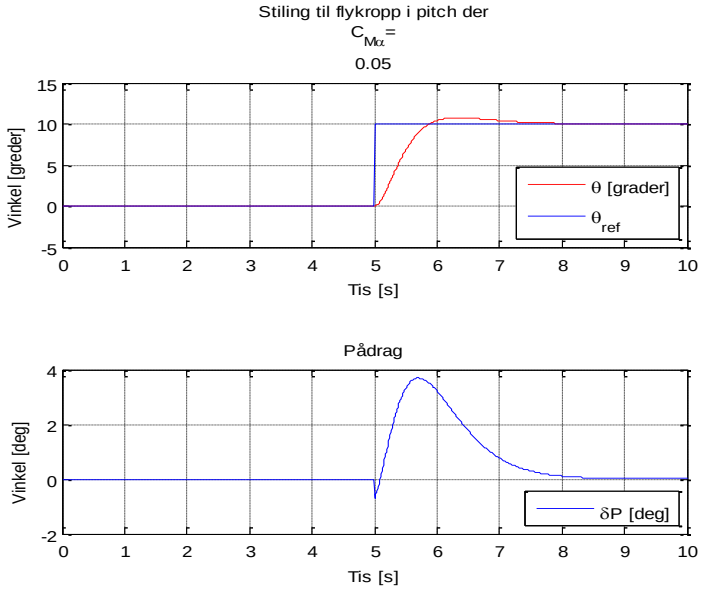
$$P_z = \begin{bmatrix} 1.1173 \\ 1.000 \\ 0.8765 \end{bmatrix} \quad (6.41)$$

Som har en pol utenfor enhetssirkelen. Denne fører til at man har et ustabilt system. Siden vi har en nullpunkt i venstre havlplan får vi et negativt rorutslag i starten av et sprang.



Figur 45: Poler og nullpunkter for pitchsystem med $C_{M\alpha} = 0.05\left[\frac{1}{deg}\right]$

Her bruker man de samme polene for det lokkede systemet og vi får følgende referanse forfølging.



Figur 46: Følging av step i referansen for pitch-systemet med $C_{M\alpha} = 0.05\left[\frac{1}{deg}\right]$

Ser at det er mulig for en servomotor å følge dette pådraget.

I disse to eksemplene ser man at vi ikke trenger inkrement kontroll, men når vi får med gravitasjon så vil det være aktuelt. Siden vi skal styre flyet etter koordinater, dermed får vi en ytre sløyfe som vil bidra til at vi ikke trenger et integralledd i den indre sløyfen for pitch-systemet.

7 Identifisering av flymodellen i pitch-systemet

I dette kapitelet skal to måter å identifisere pitch-systemet diskutere. Først skal vi bruke en rekursiv metode som bruker pådragssignalet og utgangssignalet til å lage en diskret modell av pitch-systemet, der vi har tatt utledningen fra (Wellstead, et al., 1991 ss. 71-163). Her viser det seg at vi er svært avhengig av å eksitere modellen med et spesifikt signal. Dette fører til at vi ikke kan bruke denne identifikasjonen i den adaptive regulatoren til flyet. I den andre metoden bruker vi all den kunnskapen vi har om pitch-systemet og lager en state space modell. Deretter diskretiserer vi med en rekkeutvikling og finner den diskrete transferfunksjonen som vi kan bruke på poltildelingsmetoden.

7.1 Recursive Least Square (RLS)

Vi ønsker å identifisere den diskrete transferfunksjonen for vinkelen i pitch-systemet kontinuerlig. Dette kan vi gjøre ved hjelp av Recursive least Square (RLS), Recursive Extended Least Square (RELS) eller Approximate Maximum Likelihood (AML). Hvis systemet har hvit støy bør vi bruke RLS. Hvis systemet har farget støy så bør vi bruke RELS eller AML. Siden vi ikke har implementert noe farget støy på den ulineære modellen så trenger vi ikke å bruke de mer regnekrevende algoritmene RELS og AML.

I RLS algoritmen bruker vi inn og utsignalet så fort det blir tilgjengelig. Modellen som er basert på tidligere beregninger, blir brukt til å estimere et estimert ut signal $\hat{y}(t)$. Dette blir brukt til å finne en error ($y(t) - \hat{y}(t\ ) = \epsilon(t)$), som korigerer den forige predikerte modellen $\hat{\theta}(t-1)$. Denne korrigerer den gamle modellen og vi får den nye estimerte modellen $\hat{\theta}(t)$. Med dette trenger vi ikke å lagre alle de foregående inn og ut signalene, som vi må i Least Square (LS) algoritmen.

For å komme frem til RLS algoritmen må vi se på hvordan Least Square algoritmen fungerer, deretter drive RLS algoritmen fra denne.

7.1.1 Least Square (LS)

Vi betrakter den diskrete ARX modellen, på formen

$$Ay(t) = Bu(t-1) \tag{7.1}$$

Der

$$\begin{aligned} A &= 1 + a_1 z^{-1} + \cdots + a_{n_a} z^{-n_a} \\ B &= b_0 + b_1 z^{-1} + \cdots + b_{n_b} z^{-n_b} \end{aligned} \tag{7.2}$$

Som man kan skrives på formen

$$y(t) = \varphi^{\mathrm{T}}\theta \tag{7.3}$$

Der θ er de ukjente parameterne til systemet

$$\theta = \left[a_1, a_2, \dots, a_{n_a}, b_0, b_1, \dots, b_{n_b}\right] \tag{7.4}$$

Vektoren φ^{T} inneholder inn og ut signaler

$$\varphi^{\mathrm{T}}(t) = [-y(t-1), -y(t-2), \dots - y(t-n_a), u(t-1), \dots, u(t-n_b)] \tag{7.5}$$

Når vi antar at $y(t)$ er fra den korrekte modellen får vi

$$y(t) = \varphi^{\mathrm{T}}(\mathrm{t})\hat{\theta} + \hat{\mathrm{e}}(\mathrm{t}) \tag{7.6}$$

Der $\hat{e}(t)$ er feilen på estimertoren og $\hat{\theta}(t)$ er justerbar modellparametere. Målet er å minimalisere $\hat{e}(t)$ slik at man får en korrekt modell.

$$\hat{e}(t) = \varphi^{\mathrm{T}}(\mathrm{t})\left(\theta - \hat{\theta}\right) \tag{7.7}$$

For å estimere parameterne må man ha med minst m antall likninger, der m er antall ukjente parametere. På matriseform får man da

$$\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix} = \begin{bmatrix} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \vdots \\ \varphi^{\mathrm{T}}(\mathrm{N}) \end{bmatrix} \hat{\theta} + \begin{bmatrix} \hat{\mathrm{e}}(1) \\ \hat{\mathrm{e}}(2) \\ \vdots \\ \hat{\mathrm{e}}(\mathrm{N}) \end{bmatrix} \tag{7.8}$$

$$\boldsymbol{y} = \Phi\hat{\theta} + \hat{\mathbf{e}} \tag{7.9}$$

Da kan man velge å minimalisere utrykket med least square

$$J = \sum_{t=1}^{\infty} \hat{e}^{\,2}(t) = \hat{\mathbf{e}}^{\,\mathrm{t}}\hat{\mathbf{e}} \tag{7.10}$$

For å finne least square estimatet må vi skrive om likningen over og får

$$\begin{aligned} J &= \left(\boldsymbol{y} - \Phi\hat{\theta}\right)^T\left(\boldsymbol{y} - \Phi\hat{\theta}\right) \\ &= \boldsymbol{y}^T\boldsymbol{y} - \hat{\theta}^T\Phi^T\boldsymbol{y} - \boldsymbol{y}^T\Phi\hat{\theta} + \hat{\theta}^T\Phi^T\Phi\hat{\theta} \end{aligned} \tag{7.11}$$

Setter den deriverte av $\hat{\theta}$ lik null

$$\frac{\partial J}{\partial \hat{\theta}} = -2\Phi^T\boldsymbol{y} + 2\Phi^T\Phi\hat{\theta} = 0 \tag{7.12}$$

Dette er et globalt minimum hvis (7.13) er positiv

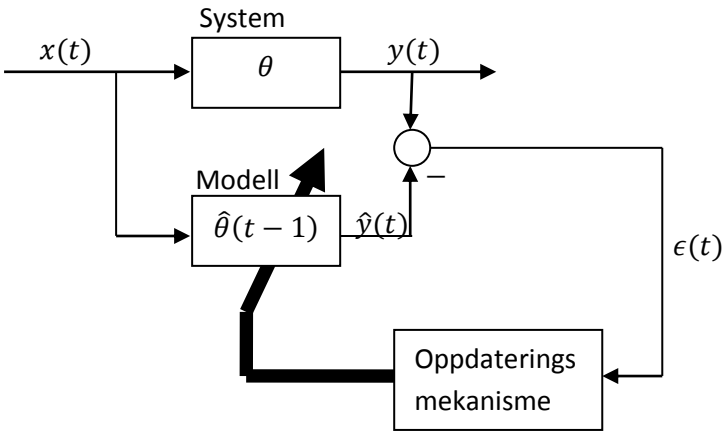
$$\frac{\partial J^2}{\partial \hat{\theta}^2} = 2(\Phi^T\Phi) \tag{7.13}$$

Least square estimatet av parametervektoren er

$$\hat{\theta} = (\Phi^T\Phi)^{-1}\Phi^T\boldsymbol{y} \tag{7.14}$$

7.1.2 Recursive least Square (RLS)

Nå skal vi drive en rekursiv metode ut i fra LS metoden der vi nå skal se på tidsrommet 1 til $t+1$. I motsetning til tidligere da vi så på tidsrommet 1 til t.



Figur 47: System med gjenkjenning

Da får man

$$\Phi(t+1) = \begin{bmatrix} \varphi^{\mathrm{T}}(1) \\ \varphi^{\mathrm{T}}(2) \\ \vdots \\ \varphi^{\mathrm{T}}(\mathrm{t}) \\ \varphi^{\mathrm{T}}(\mathrm{t}+1) \end{bmatrix} = \begin{bmatrix} \Phi(\mathrm{t}) \\ \varphi^{\mathrm{T}}(\mathrm{t}+1) \end{bmatrix}$$

$$\boldsymbol{y}(t+1) = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(t) \\ y(t+1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}(t) \\ y(t+1) \end{bmatrix} \tag{7.15}$$

Estimatet ved $t+1$ blir da gitt av

$$\hat{\theta}(t+1) = [\Phi^{\mathrm{T}}(\mathrm{t}+1)\Phi(\mathrm{t}+1)]^{-1}\Phi^{\mathrm{T}}(\mathrm{t})\mathbf{y}(\mathrm{t}+1) \tag{7.16}$$

Man ser at

$$\begin{aligned} \Phi^{\mathrm{T}}(\mathrm{t}+1)\Phi(\mathrm{t}+1) &= [\Phi^{\mathrm{T}}(\mathrm{t}) \quad \varphi(\mathrm{t}+1)] \begin{bmatrix} \Phi(t) \\ \varphi^{\mathrm{T}}(\mathrm{t}+1) \end{bmatrix} \\ &= \Phi^{\mathrm{T}}(\mathrm{t})\Phi(t) + \varphi(\mathrm{t}+1)\varphi^{\mathrm{T}}(\mathrm{t}+1) \end{aligned} \tag{7.17}$$

Nå må man finner den inverse av $\Phi^{\mathrm{T}}(\mathrm{t})\Phi(t)$ direkte uten å bygge opp den inverse matrisen. Man må i tillegg oppdatere utrykket $\Phi^{\mathrm{T}}(\mathrm{t})\mathbf{y}(\mathrm{t}+1)$. Da skriver man utrykket på samme måte som likningen over og får

$$\Phi^{\mathrm{T}}(\mathrm{t})\mathbf{y}(\mathrm{t}+1) = \Phi^{T}(t)\boldsymbol{y}(t) + \varphi(\mathrm{t}+1)\mathrm{y}(\mathrm{t}+1) \tag{7.18}$$

Introduserer noen nye variable slik at det blir lettere og regne ut

$$P(t) = [\Phi^{\mathrm{T}}(\mathrm{t})\Phi(t)]^{-1}$$
$$B(t) = \Phi^{T}(t)\boldsymbol{y}(t) \tag{7.19}$$

Hvor

$$\hat{\theta}(t+1) = P(t+1)B(t+1)$$
$$\hat{\theta}(t) = P(t)B(t) \tag{7.20}$$

Da blir

$$P^{-1}(t+1) = P^{-1}(t) + \varphi(\mathrm{t}+1)\varphi^{\mathrm{T}}(\mathrm{t}+1)$$
$$B(t+1) = B(t) + \varphi(\mathrm{t}+1)\mathrm{y}(\mathrm{t}+1) \tag{7.21}$$

Det kritiske her er å finne $P(t+1)$. Dette kan man gjøre med invers matrise lemma, siden kovarians matrisen er ikke- singulær og rektangulær kan vi bruke det inverse matrise lemma.

$$(A+BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1} \tag{7.22}$$

Der vi setter

$$A = P^{-1}(t), \qquad C = 1, \qquad B = \varphi(\mathrm{t}+1), \qquad \mathrm{D} = \varphi^{\mathrm{T}}(\mathrm{t}+1) \tag{7.23}$$

Som gir

$$P(t+1) = P(t)\left(I_m - \varphi(\mathrm{t}+1)\left(1 + \varphi^{\mathrm{T}}(\mathrm{t}+1)\mathrm{P}(\mathrm{t})\varphi(\mathrm{t}+1)\right)^{-1}\varphi^{\mathrm{T}}(\mathrm{t}+1)\mathrm{P}(\mathrm{t})\right) \tag{7.24}$$

Her ser man at vi ikke trenger å invertere matrisen, bare en skalar.

Man kan skrive feilen på formen

$$\varepsilon(t+1) = y(t+1) - \varphi^{\mathrm{T}}(\mathrm{t}+1)\hat{\theta}(\mathrm{t}) \tag{7.25}$$

Når man setter dette inn for $y(t+1)$ i likning (7.21) får man

$$B(t+1) = B(t) + \varphi(\mathrm{t}+1)\varphi^{\mathrm{T}}(\mathrm{t}+1)\hat{\theta}(\mathrm{t}) + \varphi(\mathrm{t}+1)\varepsilon(t+1) \tag{7.26}$$

Setter man inn utrykket for $B(t)$ og $B(t+1)$ fra likning (7.20) får man

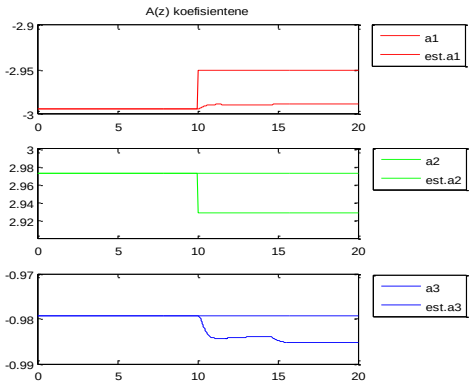
$$\hat{\theta}(t+1) = \hat{\theta}(t) + P(t+1)\varphi(\mathrm{t}+1)\varepsilon(t+1) \tag{7.27}$$

Dette gir RLS algoritmen:

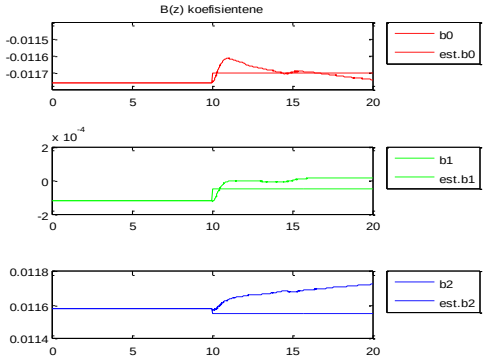
1. $\Phi(\mathrm{t}+1) = [-y(t-1) \quad \dots \quad -y(t-n_a) \quad u(t-1) \quad \dots \quad u(t-n_b-1)]^T$
2. $\epsilon(t+1) = y(t) - \varphi^{\mathrm{T}}(\mathrm{t})\hat{\theta}(\mathrm{t}-1)$
3. $P(t+1) = P(t)\left(I_m - \varphi(\mathrm{t}+1)\left(1 + \varphi^{\mathrm{T}}(\mathrm{t}+1)\mathrm{P}(\mathrm{t})\varphi(\mathrm{t}+1)\right)^{-1}\varphi^{\mathrm{T}}(\mathrm{t}+1)\mathrm{P}(\mathrm{t})\right)$
4. $\hat{\theta}(t+1) = \hat{\theta}(t) + P(t+1)\varphi(\mathrm{t}+1)\varepsilon(t+1)$

7.1.3 Tester RLS algoritmen

Vi tester RLS algoritmen på transferfunksjonen i pitch-systemet og ser at den ikke følger forandringen i parameterne. Dette er som forventet siden RLS algoritmen ikke er beregnet til å følge forandringer i parameterne. Grunnen til dette er at kovariansmatrisen ”sovner”, blir svært liten og greier ikke å følge med.



Figur 48: Gjenkjenning av polene til forenklet pitch-system med RLS



Figur 49: Gjenkjenning av nullpunktene til forenklet pitch-system med RLS

Her har vi brukt en firekantpuls med amplitude på 10^0 og en periode på $10\ s$. For at RLS algoritmen skal fungere trenger vi å legge på hvit støy på inngangen.

7.2 Følge parameterendringer med RLS metoden

Den RLS algoritmen som er nevnt ovenfor er egnet til å finne konstante prosessparametere. Siden vi skal følge varierende prosessparametere må vi modifisere algoritmen over. Dette kan vi gjøre på to måter Random Walk eller Forgetting Factor.

7.2.1 Random Walk

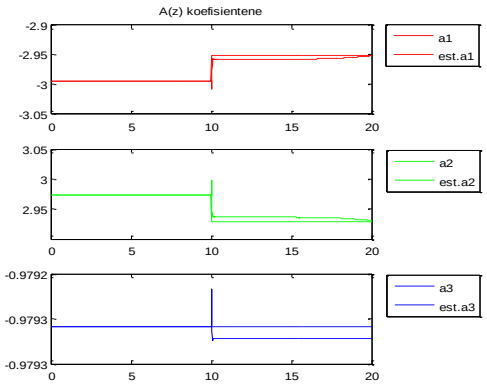
En symmetrisk positiv bestemt matrise R kan bli plusset på kovariansmatrisen $P(t)$ for å forhindre at den blir for liten. Denne parameteren kan vi legge til når vi vet at det blir en forandring i systemet, eller la den være der kontinuerlig hvis vi ikke vet når forandringen kommer. Kovariansmatrisen blir som følger

$$P(t) = P(t-1)\left[\mathrm{I_m} - \frac{\varphi(\mathrm{t})\varphi(\mathrm{t})^T P(t-1)}{1+\varphi^{\mathrm{T}}(\mathrm{t})\mathrm{P}(\mathrm{t}-1)\varphi(\mathrm{t})}\right] + R \tag{7.28}$$

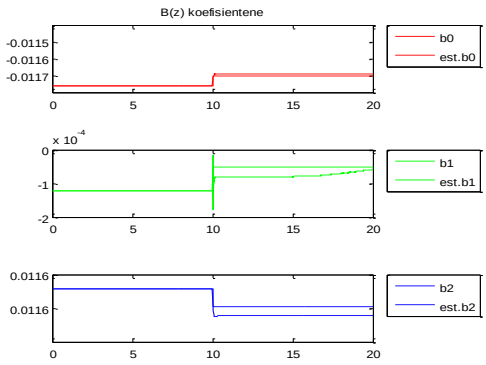
Der R oftest velges til å være diagonal og en god antagelse på $R = rI_m$ der $r = 0.1 \rightarrow 0.5$. På tross av dette velger vi en $R = diag(r_1, \dots, r_6)$ der r_n er utenfor $r = 0.1 \rightarrow 0.5$. Dette gjør vi for å få raskest mulig tilpassing av parameterne til systemet. Her er det cut and tray prinsippet som gjelder for å finne disse 6 parameterne. Det vi må huske på er at r må være positiv siden $P(t)$ er kovarians matrisen.

7.2.1.1 Tester Random Walk på pitch transferfunksjon

Vi bruker transferfunksjonen i pitch-systemet, da kan vi lettere forandre systemparameterne på en kontrollert måte når vi skal justere R parameteren. Med litt ”cut end tray” får vi følgende $R = diag(500000, 500000, 0, 0.09, 60, 0.01)$



Figur 50: Gjenkjenning av polene til forenklet pitch-system med Random Walk



Figur 51: Gjenkjenning av nullpunktene til forenklet pitch-system med Random Walk

Her har vi brukt en firekantpuls med amplitude på 10^0 og en periode på $10\ s$. For at RLS algoritmen skal fungere må vi legge på hvit støy på inngangen, denne har en min maks verd på ± 0.01.

7.2.1 Forgetting Factor

Den andre muligheten er å bruke Forgetting Factor. Her har man en faktor λ som er normalt mellom $0.95 - 0.99$. Som brukes til å redusere avhengigheten av tidligere informasjon om prosessen. Ideen kan best bli beskrevet med å se på kostfunksjonen til normal least square som vi minimaliserer ved hvert tidskritt

$$J_t = \sum_{i-1}^{t} \hat{e}^2(i) \tag{7.29}$$

Forgetting Factor bruker en annen vekting og får den modifiserte kost funksjonen

$$\bar{J}_t = \sum_{i-1}^{t} \lambda^{t-1} \hat{e}^2(i) \tag{7.30}$$

Som kan skrives på diskret form

$$\bar{J}_t = \lambda \bar{J}_{t-1} + \hat{e}^2(t) \tag{7.31}$$

Her ser man at λ reduserer verdien på de gamle dataene. Når vi bruker den nye kostfunksjonen får man den modifiserte kostfunksjonen

$$\hat{\theta} = [X^T(t)\Lambda(t)X(t)]^{-1}X^T(t)\Lambda(t)y(t) \tag{7.32}$$

Hvor $\Lambda = diag(\lambda^{t-1}, \lambda^{t-2}, \dots, \lambda^2, 1)$

Med dette kan man finne den modifiserte kovariansmatrisen på samme måte som man gjorde for Rekursive minste kvadraters metode.

$$P^{-1}(t) = \lambda P^{-1}(t-1) + \varphi(t)\varphi^T(t) \tag{7.33}$$

$$P(t) = \lambda^{-1}P(t-1)\left[\mathrm{I_m} - \frac{\varphi(\mathrm{t})\varphi(\mathrm{t})^T P(t-1)}{\lambda + \varphi^{\mathrm{T}}(\mathrm{t})\mathrm{P}(\mathrm{t}-1)\varphi(\mathrm{t})}\right] \tag{7.34}$$

Her ser man at vi har en ulempe, alle elementene i kovariansmatrisen er skalert likt. Dette er det mulig å komme rundt, men det går vi ikke inn på her.

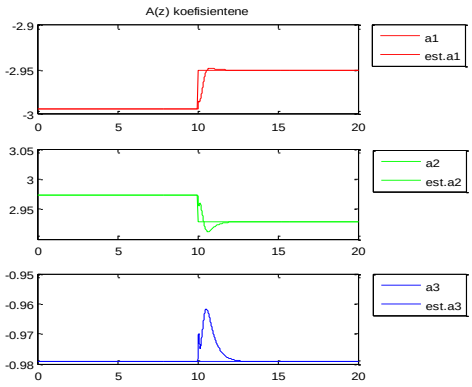
For at ikke forandringen i prosessparameterne blir for store, er vi avhengig av å bruke et 1. ordens lavpass filter (Haugen, 2003a ss. 150-158)

$$\frac{y(t)}{u(t)} = \frac{\omega_b T_s(1+z^{-1})}{\omega_b Ts + 2 + (\omega_b T_s - 2)z^{-1}} \tag{7.35}$$

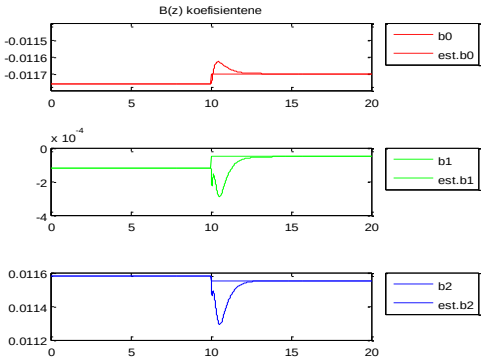
Her har vi testet ut noen forskjellige båndbredder og finner ut at $\omega_b = 2-3$ er et godt valg. Siden vi har lagt inn ett lavpass filter slår vi ikke på RLS algoritmen før LP filteret får tid til å henge med. LP filteret bruker ca. 5 sekunder. Dette LP- filteret fører til at vi unngår å få et ustabilt system.

7.2.2.1 Tester Forgetting Factor på Pitch transferfunksjon

Tester ut med en firekantpuls i referansen med amplitude på 10^0 og en periode på $10\ s$.
For at RLS algoritmen skal fungere trenger vi å legge på hvit støy på inngangen, denne har en min/ maks verd på ± 0.01.



Figur 52: Gjenkjenning av polene til forenklet pitch-system med Forgetting Factor



Figur 53: Gjenkjenning av nullpunktene til forenklet pitch-system med Forgetting Factor

Ser at vi har en tilfredsstillende forfølging av prosessparameterne. Fordelen med Forgetting Factor fremfor Random Walk er at den er mye lettere å justere inn en Random Walk.

7.2.3 Konstant trace algoritme

Hvis φ bringer lite ny informasjon vil $P(t) \approx P(t-1)$. Da vil kovariansmatrisen $P(t)$ øke for hvert tidskritt og vi får "*estimator wind-up*". Dette kan vi forhindre med å bruke en R matrise som i Random Walk, der R velges slik at tracen til $P(t)$ holdes konstant. En annen måte å unngå "*estimator wind-up*" er å bruke en varierende Forgetting Factor.

Her legger vi til en $R(t)$ matrise som holder tracen av kovariansmatrisen $P(t)$ konstant. Dette får man til med

$$\mathrm{tr}(R(t)) = \frac{\varphi(\mathrm{t})^T P^2(t-1)\varphi(\mathrm{t})}{1+\varphi^{\mathrm{T}}(\mathrm{t})\mathrm{P}(\mathrm{t}-1)\varphi(\mathrm{t})} \tag{7.36}$$

Som er tilfredsstilt ved

$$R(t) = m^{-1} tr\big(R(t)\big) I_m \tag{7.37}$$

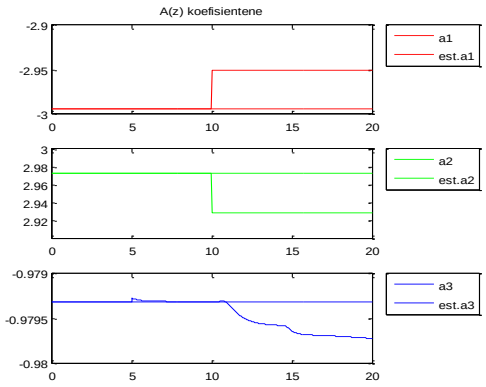
Denne algoritmen eliminerer "*estimator wind-up*" og gir estimator algoritmen følgemuligheter. Hver oppmerksom på at hvis vi har lite ny informasjon vil $P(t)$ bli nesten singulær og $P(t)\varphi(\mathrm{t}+1)$ vil nesten bli null. Man får dermed den fulle algoritmen

$$\begin{gathered}
\Phi(\mathrm{t}+1) = [-y(t-1) \quad \dots \quad -y(t-n_a) \quad u(t-1) \quad \dots \quad u(t-n_b-1)]^T \\
\epsilon(t+1) = y(t) - \varphi^{\mathrm{T}}(\mathrm{t}+1)\hat{\theta}(\mathrm{t}) \\
K(t) = \frac{P(t)\,\varphi(\mathrm{t}+1)}{1+\varphi^{\mathrm{T}}(\mathrm{t}+1)\mathrm{P}(\mathrm{t})\varphi(\mathrm{t}+1)} \\
\hat{\theta}(t+1) = \hat{\theta}(t) + K(t)\epsilon(t+1) \\
\mathrm{tr}\big(R(t)\big) = \frac{\varphi(\mathrm{t}+1)^T P^2(t)\varphi(\mathrm{t}+1)}{1+\varphi^{\mathrm{T}}(\mathrm{t}+1)\mathrm{P}(\mathrm{t})\varphi(\mathrm{t}+1)} \\
R(t) = m^{-1}\mathrm{tr}\big(R(t)\big) I_m \\
P(t+1) = P(t)\left[\mathrm{I_m} - \frac{\varphi(\mathrm{t}+1)\varphi(\mathrm{t}+1)^T P(t)}{1+\varphi^{\mathrm{T}}(\mathrm{t}+1)\mathrm{P}(\mathrm{t})\varphi(\mathrm{t}+1)}\right] + R(t)
\end{gathered} \tag{7.38}$$

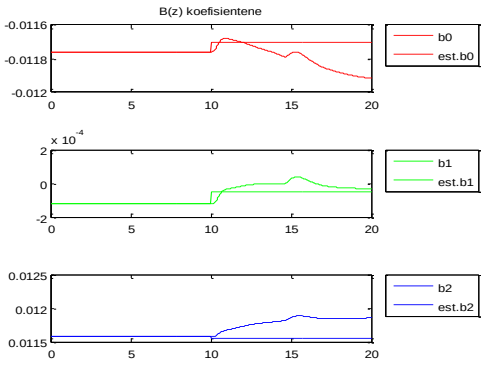
Denne algoritmen har den fordelen at den greier en større forandring i inn signalets amplitude. Den mister ikke følgeegenskapene hvis vi legger på en støy på styresignalet som ikke vil påvirke flyet unngår vi at kovariansmatrisen går mot singulær tilstand hvis vi har lite ny informasjon, holder konstant referanse.

7.2.3.1 Tester konstant trace algoritme på pitch transferfunksjon

Vi prøver oss litt frem og finner en initialtilstand på kovariansmatrisen på $I_m * 100$ som gir oss følgende koeffisienter



Figur 54: Gjenkjenning av polene til forenklet pitch-system med Konstant trace



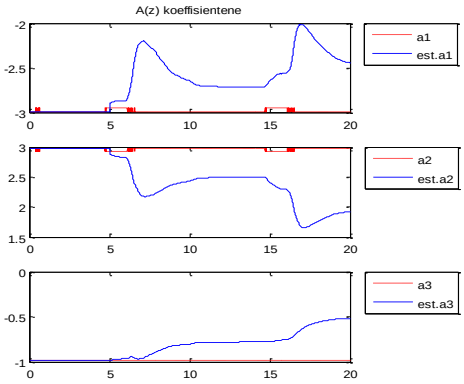
Figur 55: Gjenkjenning av nullpunktene til forenklet pitch-system med Konstant trace

Som vi ser på figuren over får vi en svært dårlig forfølging av systemparameterne, da systemet har store forandringer i $A(z)$ koeffisientene og svært små forandringer i $B(z)$ koeffisientene. Så hvis denne algoritmen skal fungere må man bruke en annen fordeling av konstant trace.

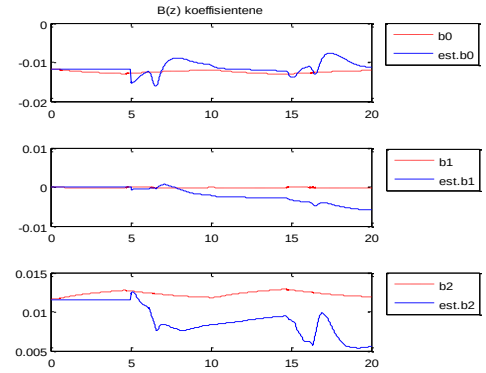
7.2.4 Random Walk på det ulineære systemet

Vi kjører simulatoren med de tre indre sløyfene og ser om vi greier å gjenkjenne det ulineære systemet. Her har vi en firkantpuls på pitch referansen på 17^0 og perioden på $10[s]$. Da får vi et sprang i pitch-systemet når angrepsvinkel er under -8^0, som er i tidsrommet ca.5-6 og 15-16.

Vi får følgende gjenkjenning



Figur 56: Gjenkjenning av polene til fult pitch-system med Random Walk

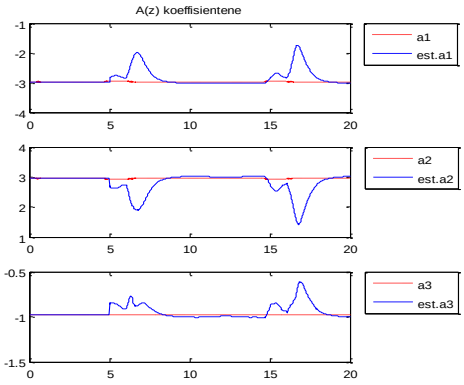


Figur 57: Gjenkjenning av nullpunktene til fult pitch-system med Random Walk

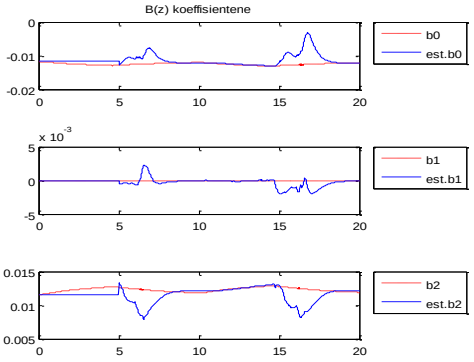
Som vist på figuren over greier vi ikke å gjenkjenne systemet på en tilfredsstillende måte.

7.2.5 Forgetting Factor på det ulineære systemet

Bruker samme modell som vi brukte for Random Walk, og får følgende gjenkjenning



Figur 58: Gjenkjenning av polene til fult pitch-system med Forgetting Factor



Figur 59: Gjenkjenning av nullpunktene til fult pitch-system med Forgetting Factor

Ser på figuren over at vi får noe bedre gjenkjenning, men som vi skal se i kapitel 8 vil denne gjenkjenningen føre til et ustabilt system når den blir brukt i sammen med en adaptiv regulator.

7.3 Transferfunksjon basert identifisering

Her bruker vi modellen vi fant for pitch-systemet. Det viktigste her er å få ned regnetiden på beregningen av diskretiseringen av systemmatrisen A og syringsmatrisen B. Den enkleste måten å beregne Φ på er ved rekkeutvikling av $e^{A\Delta t}$.

$$\Phi = e^{A\Delta t} = \sum_{n=0}^{\infty} \frac{1}{k!} \Delta t^n A^n \tag{7.39}$$

Ved å anta at pådraget u er konstant i diskretiseringsintervallet, kan Λ beregnes ved å løse integralet

$$\Lambda = \left(\int_0^{\Delta t} e^{A\tau}\, d\tau \right) B$$

$$= \left(\sum_{n=0}^{\infty} \frac{1}{(k+1)!} \Delta t^{(n+1)} F^n \right) B \tag{7.40}$$

$$= \left(\sum_{n=0}^{\infty} \frac{1}{(k+1)!} \frac{1}{\mathrm{k!}}\, \Delta t^n F^n \right) B\Delta t$$

Siden transferfunksjonen for pitch-systemet bare inneholder Φ, Δ og C kan vi beregne koeffisientene til transferfunksjonen

$$H(z) = \frac{\begin{array}{c}(-C_2\Delta_2)z^2 + \big(C_2(\Delta_2\Phi_{11} + \Delta_2\Phi_{33} - \Delta_1\Phi_{21} - \Delta_3\Phi_{23})\big)\mathrm{z} \\ + \Big(\mathrm{C}_2\big(\Delta_1(\Phi_{21}\Phi_{33} - \Phi_{31}\Phi_{23}) + \Delta_2(\Phi_{13}\Phi_{31} - \Phi_{11}\Phi_{33}) + \Delta_3(\Phi_{11}\Phi_{23} - \Phi_{21}\Phi_{13})\big)\Big)\end{array}}{\begin{array}{c}-z^3 + (\Phi_{11} + \Phi_{22} + \Phi_{33})z^2 + (\Phi_{12}\Phi_{21} + \Phi_{13}\Phi_{31} + \Phi_{23}\Phi_{32} - \Phi_{11}(\Phi_{22} + \Phi_{33}) - \Phi_{22}\Phi_{33})z \\ + \big(\Phi_{11}(\Phi_{22}\Phi_{33} - \Phi_{23}\Phi_{32}) + \Phi_{12}(\Phi_{31}\Phi_{23} - \Phi_{21}\Phi_{33}) + \Phi_{13}(\Phi_{21}\Phi_{32} - \Phi_{22}\Phi_{31})\big)\end{array}} \tag{7.41}$$

Siden

$$H(z) = \frac{b_0 z^2 + b_1 z + b_2}{z^3 + a_1 z^2 + a_2 z + a_3} \tag{7.42}$$

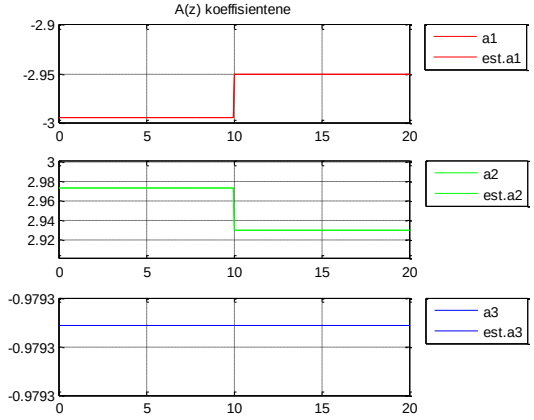
Får vi

$$\begin{aligned}
b_0 &= -C_2\Delta_2 \\
b_1 &= C_2(\Delta_2\Phi_{11} + \Delta_2\Phi_{33} - \Delta_1\Phi_{21} - \Delta_3\Phi_{23}) \\
b_2 &= C_2\big(\Delta_1(\Phi_{21}\Phi_{33} - \Phi_{31}\Phi_{23}) + \Delta_2(\Phi_{13}\Phi_{31} - \Phi_{11}\Phi_{33}) + \Delta_3(\Phi_{11}\Phi_{23} - \Phi_{21}\Phi_{13})\big) \\
a_1 &= \Phi_{11} + \Phi_{22} + \Phi_{33} \\
a_2 &= \Phi_{12}\Phi_{21} + \Phi_{13}\Phi_{31} + \Phi_{23}\Phi_{32} - \Phi_{11}(\Phi_{22} + \Phi_{33}) - \Phi_{22}\Phi_{33} \\
a_3 &= \Phi_{11}(\Phi_{22}\Phi_{33} - \Phi_{23}\Phi_{32}) + \Phi_{12}(\Phi_{31}\Phi_{23} - \Phi_{21}\Phi_{33}) + \Phi_{13}(\Phi_{21}\Phi_{32} - \Phi_{22}\Phi_{31})
\end{aligned} \tag{7.43}$$

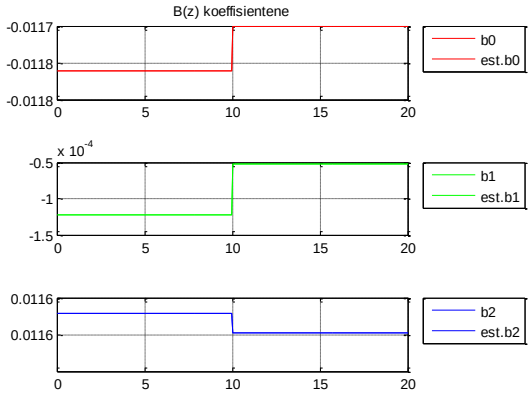
Med dette kan vi lage en kontinuerlig oppdatering av systemparameterne med måling av angrepsvinkelen α som sier oss hvilken verdi vi skal ha på aerodynamiske koeffisienten C_M. Måler vi den totale hastigheten kan vi også inkludere denne i beregningen av identifiseringen.

7.3.1 Teorien på pitch gjenkjenning testes ut

Nå bruker vi det lineariserte systemet for enklest mulig å teste om gjenkjenningsalgoritmen stemmer. Grunne til dette er at den ulineære modellen er mye mer ustabil enn den lineære og dermed er det mye vanskeligere å få en angrepsvinkel over 8 grader ved 10 sekunder.



Figur 60: Gjenkjenning av polene med diskretisering av pitch transferfunksjon



Figur 61: Gjenkjenning av nullpunktene med diskretisering av pitch transferfunksjon

Her ser vi at vi greier å følge parameterne eksakt uten noen forstyrrelse. Bakdelen her er at vi er avhengig av at de aerodynamiske dataene er riktig. Den krever også en litt større prosessor, men det vil ikke skape noe problem så lenge vi ikke tar med alt for mange ledd i rekkeutviklingen for diskretiseringen. Det er heller ikke nødvendig siden rekkene konvergerer raskt.

8 Adaptiv regulering

Hovedleddet i å implementere en adaptiv regulator ligger i gjenkjenne systemmodellen. Bruker vi identifikasjonsmetodene vi gikk igjennom i kapittel 7 og poltildelingsregulatoren fra kapitel 6 kan vi lage en regulator som justerer seg etter den totale hastigheten V_T og angrepsvinkelen α.

Slik modellen er, har pitch-systemet to transferfunksjoner, en for angrepsvinkler mellom 0-8 grader og en fra 8 grader og oppover, med en konstant V_T. Siden vi har en varierende V_T vil systemet være kontinuerlig varierende.

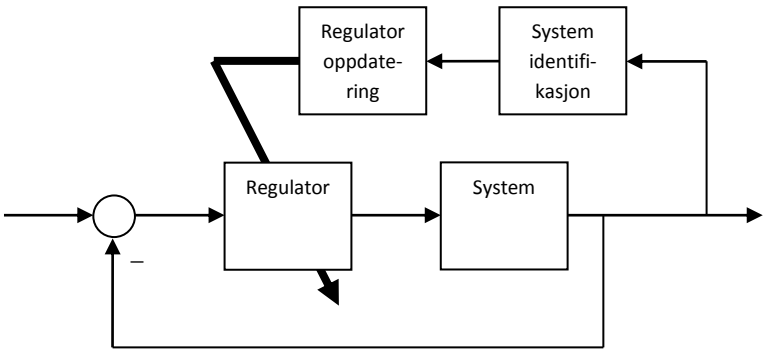
Målet er å lage en generell algoritme der vi følger ett kontinuerlig varierende system. De første metodene som blir vurdert er basert på Recursive Least Square (RLS) metoden, der vi har utleder en rekursiv metode ut fra Least Square (LS). Det andre prinsippet bruker det vi vet om systemet og de aerodynamiske koeffisientene som er forhåndsmålte via fysiske målinger eller vindtunellmålinger. Vi kan også oppdrive disse ved CFT analyse. Siden de aerodynamiske koeffisientene avhenger av angrepsvinkelen og total hastighet må vi regne ut den diskrete transferfunksjonen kontinuerlig. Noe som kan krever mange regneoperasjoner.

8.1 Adaptiv regulator med RLS

Vi bruker RLS med Random Walk/Forgetting Factor og poltildelingsregulator på det lineære systemet og deretter implementerer det på den ulineære fly modellen med variabel aerodynamisk koeffisient C_M. Vi ser om gjenkjenningen vi gjorde i kapitel 7 var god nok.

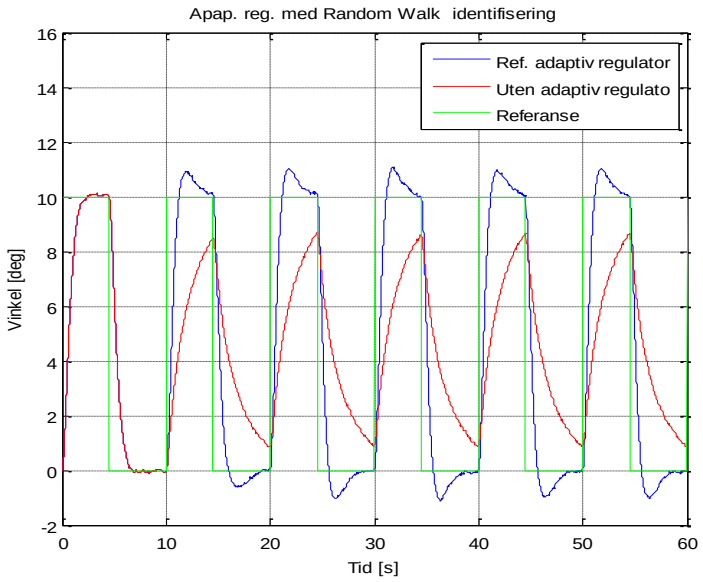
8.1.1 Adaptiv regulator på variabel lineær modell og Random Walk

Som vist på tegningen så oppdaterer RLS algoritmen modellparameterne, slik at poltildelings metoden kan bruke disse nye parameterne til å beregne en ny regulator for vært tidskritt.



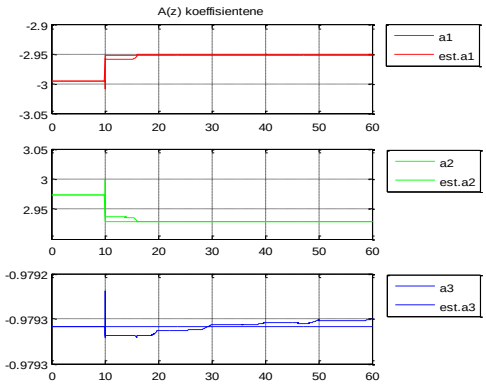
Figur 62: Blokkdiagram av adaptiv regulator

Vi simulerer modellen med Random Walk prinsippet for å identifisere prosessen, og kan dermed se forandringen på referanseforfølgingen med og uten adaptiv regulator. Her skifter vi modell ved $10\ [s]$, slik at vi får systemet som gjelder når angrepsvinkelen er over 8 grader. Den sanne vindhastigheten er konstant på $200\ [^m/_s]$.

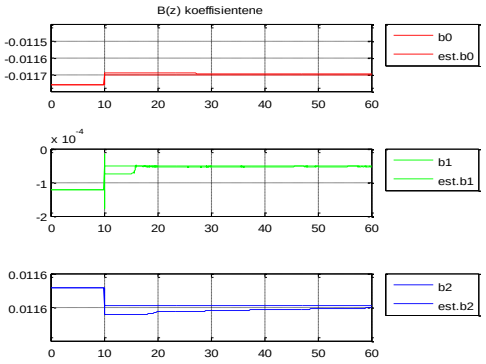


Figur 63: Pitch-vinkelen med og uten adaptiv regulator der vi har et skifte når pitch-vinkelen er konstant og man bruker Random Walk identifikasjon

Her er man nødt til å legge på hvit støy på inngangen slik at det er mulig å gjenkjenne systemet. Vi får følgende gjenkjenning når vi har oppdatering av regulator parameterne.



Figur 64: Gjenkjente poler med Random Walk på lineært system



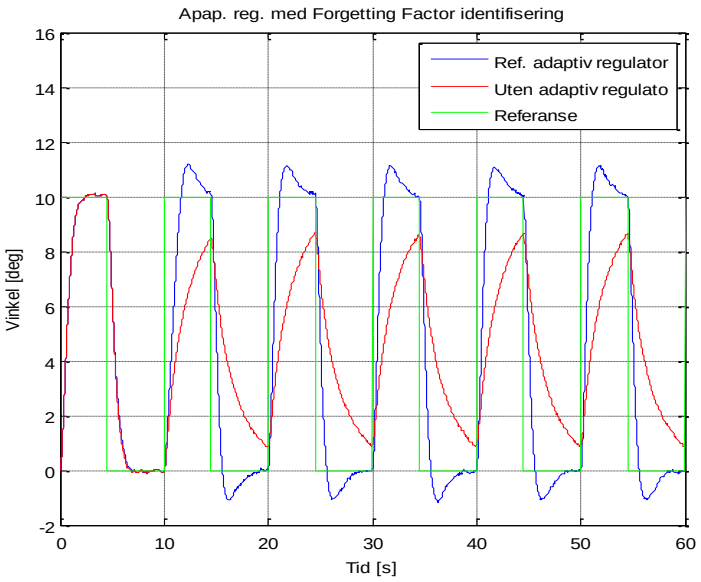
Figur 65: Gjenkjente nullpunkter med Random Walk på lineært system

Her har vi en tilfredsstillende gjenkjenning, den er rask og nøyaktig, men har vi et skifte når pitch-vinkelen er varierende får vi et ustabilt system.

8.1.2 Adaptiv regulator på variabel lineær modell og Forgetting Factor

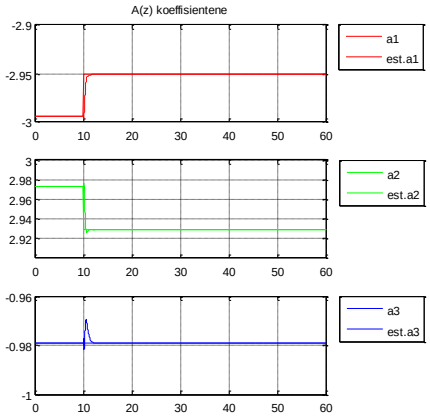
I kapitel 7 ble det vist at RLS metoden med Forgetting Factor var litt tregere enn Random Walk metoden for det lineare systemet. Siden RLS med Forgetting Factor gir en bedre gjenkjenning på et større amplitudespekter på referansen, er det fornuftig å prøve ut denne også.

Tester den adaptive algoritmen på samme måte som vi gjorde over og får som vist på figuren under.

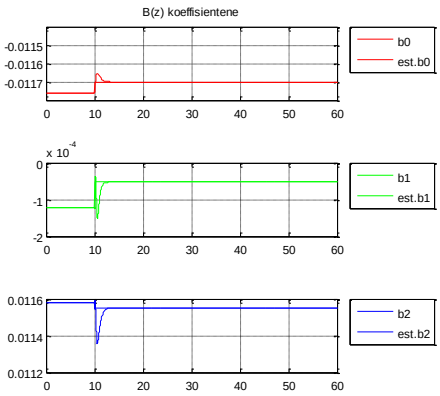


Figur 66: Pitch-vinkelen med og uten adaptiv regulator der vi har et skifte når pitch-vinkelen er konstant og man bruker Forgetting Factor identifikasjon

Vi må også her legge på hvit støy for å greie å drive frem parameterne til systemet og får følgende gjenkjenning når vi har adaptiv regulator.



Figur 67: Gjenkjente poler med Forgetting Factor på lineært modell



Figur 68: Gjenkjente nullpunkter med Forgetting Factor på lineært modell

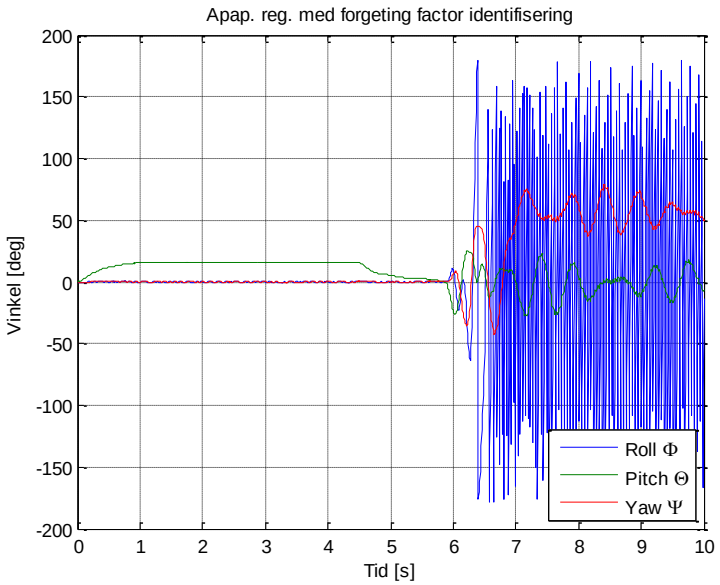
Her gir RLS med Forgetting Faktor bedre gjenkjenning enn Random Walk, den er heller ikke så avhengig av når forandringen i modellen skjer.

Problemet med disse to metodene er hvis forandring i systemet skjer når pitch-vinkelen ikke er konstant, da vil vi få et ustabilt system. Dette kommer av at vi får en forandring i pitch-vinkelen når skifte skjer og dermed vil gjenkjenningen bli feil og systemet blir ustabilt.

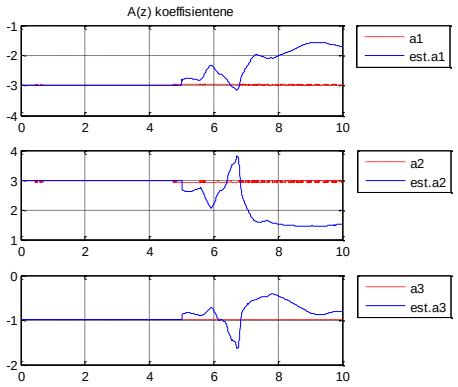
8.1.3 Adaptiv regulator på ulineær flymodell og Forgetting Factor

Man bruker de tre indre sløyfene på den ulineære modellen for å se om gjenkjenning av pitch-systemet til flyet er god nok. Vi får et ustabilt system. Dette er ikke uventende siden nullpunktene er nesten null og dermed vil en liten gal forandring i nullpunktene føre til en stor forandring i

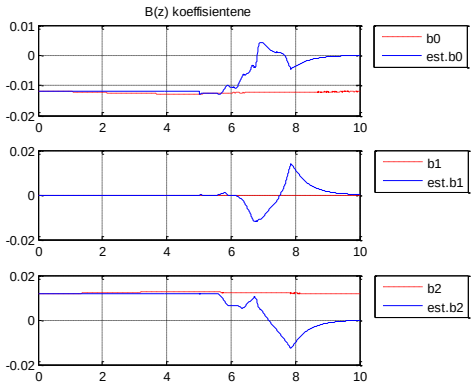
regulatorparameterne F og G. Dermed må vi gjenkjenne systemet på en annen måte eller utvikle ny og bedre RLS metode.



Figur 69: Adaptiv regulator på ulineær flymodell og Forgetting Factor til å gjenkjenne pitch systemet, med en puls på referansen på 17^0 og 0^0 med en periode på 10 s i pitch. Identifikasjonen blir satt på ved tiden 5 s.



Figur 70: Gjenkjente poler med Forgetting Factor på ulineær modell



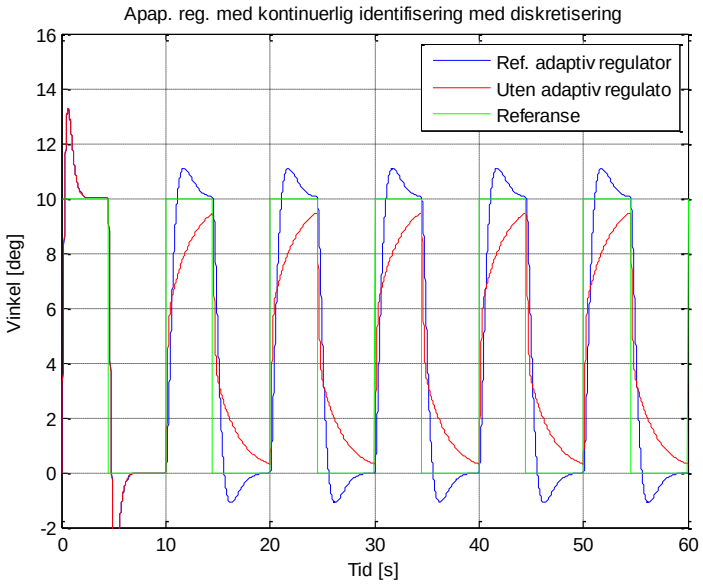
Figur 71: Gjenkjente nullpunkter med Forgetting Factor på ulineær modell

8.2 Adaptiv regulator basert på diskre transferfunksjon

Vi bruker diskre transferfunksjon til å gjenkjenne pitch systemet som beskrevet i kapitel 7 og bruker poltildelingsmetoden som beskrevet i kapitel 6. Vi tester først om dette fungerer på en lineær modell og deretter bruker vi den på indre sløyfe for pitch. Til slutt kjører vi en full simulasjon der vi ser at den adaptive regulatoren gir ett bedre resultat enn uten adaptiv regulator.

8.2.1 Adaptiv regulator på variabel lineær modell basert på diskre Tf[7]

Nå trenger man ikke lenger å legge på hvit støy for å gjenkjenne systemet og vi er heller ikke avhengige hva å ha riktig amplitude på referansesignalet for å greie å gjenkjenne systemet.

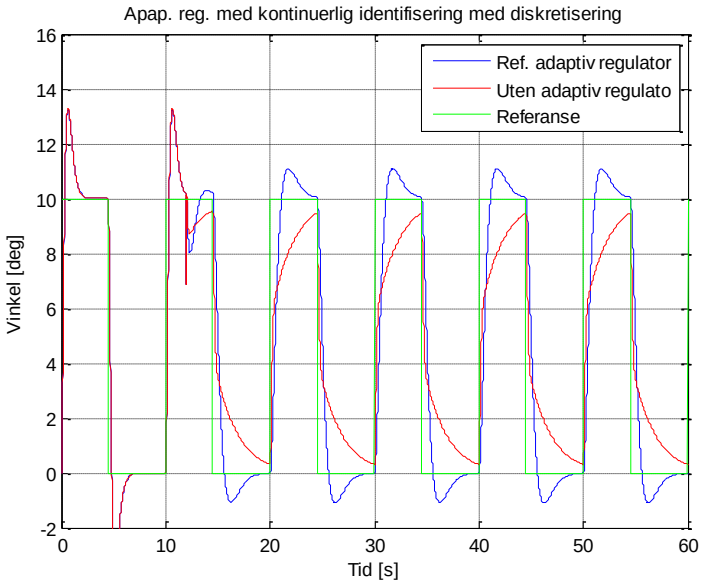


Figur 72: Pitch-vinkelen med og uten adaptiv regulator der vi har et skifte når pitch-vinkelen er konstant

Som vi ser i figuren over får vi et bra adaptivt system hvis vi skifter system når vi har en konstant pitch-vinkel. Når forandringen av systemet skjer under varierende pitch-vinkel får man en stor forandring i pitch-vinkelen som vi ser på figuren under. Derfor velger vi å ha to sett med regulatorpoler, henholdsvis når $C_{M\alpha} = 0.05$ og når $C_{M\alpha} = -0.1$. Vi tok i utgangspunkt i polene vi fant i kapittel 6 og modifiserte disse

$$T_p = (s - 0{,}9602)(s - 0{,}9602)(s - 0{,}9502)$$
$$T_{p1} = (s - 0{,}9802)(s - 0{,}9802)(s - 0{,}9802) \quad (8.1)$$

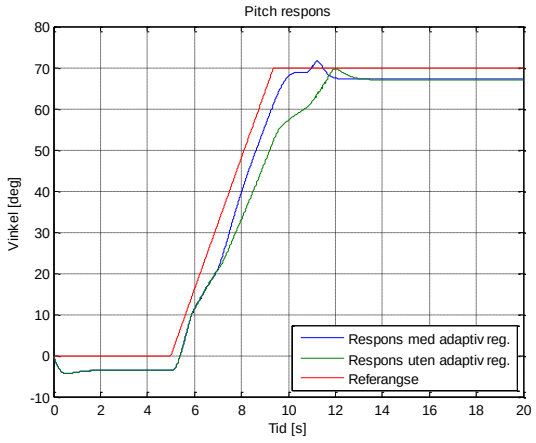
[7] Trasferfunksjon



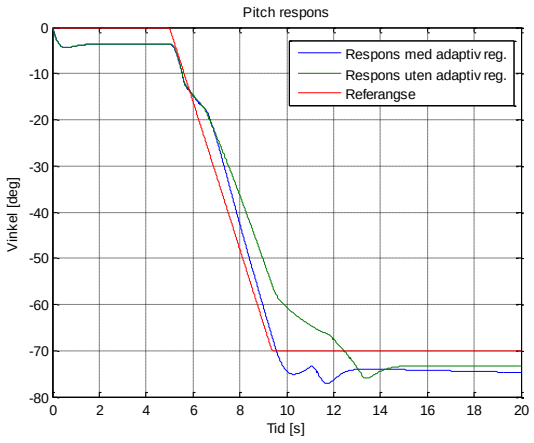
Figur 73: Pitch-vinkelen med og uten adaptiv regulator der vi har et skifte av systemmodellen når pitch-vinkelen ikke er konstant

8.2.2 Adaptiv regulator på indre sløyfe, ulineær modell

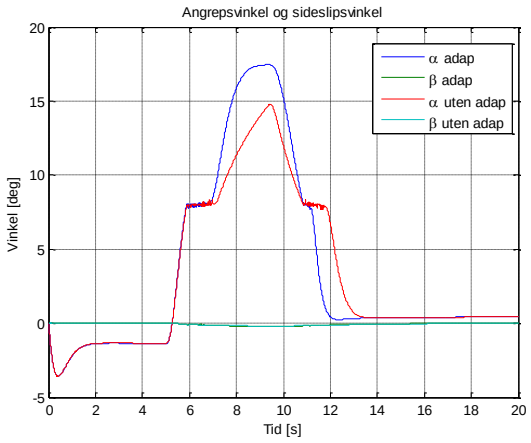
Setter inn den adaptive regulatoren i den ulineære modellen med de samme polene som vi fant for det lineære systemet.



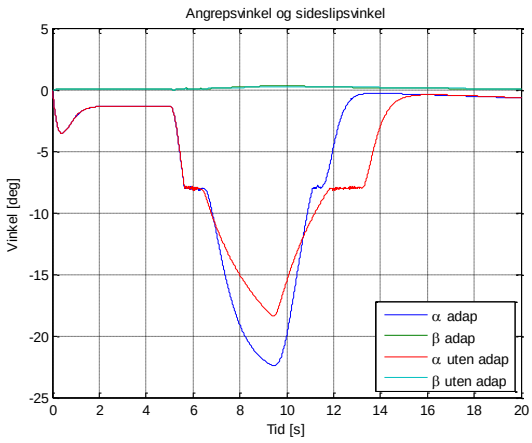
Figur 74: Plot av referanse på 70^0 i pitch til den ulineære flymodellen med og uten adaptiv regulator.



Figur 75: Plot av referanse på -70^0 i pitch til den ulineære flymodellen med og uten adaptiv regulator.

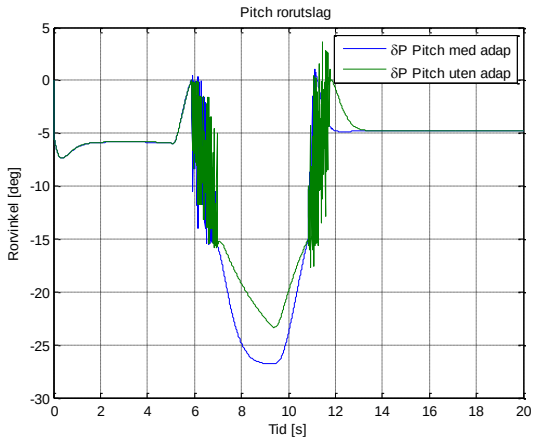


Figur 76: Plot av angrepsvinkelen med en referanse i pitch 70^0

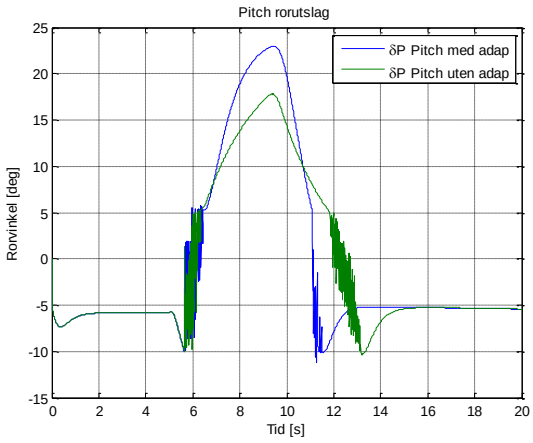


Figur 77: Plot av angrepsvinkelen med en referanse i pitch på -70^0

Ser at vi får en forbedring av referansesignalet når angrepsvinkelen er over 8^0 og under -8^0. Jo lenger man har en angrepsvinkel over 8^0 og under -8^0 jo mer tjener man på å ha adaptiv regulator. Hvis vi ser på det totale rorutslaget i pitch med og uten adaptiv regulator ser vi at begge har urolig rorutslag når vi hopper mellom det ustabile og stabile systemet og motsatt.



Figur 78: Rorutslag for samlet pitch rorutslag med en referanse i pitch på 70^0

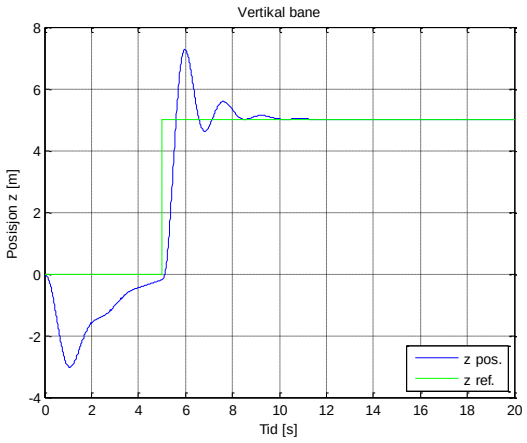


Figur 79: Rorutslag for samlet pitch rorutslag med en referanse i pitch på -70^0

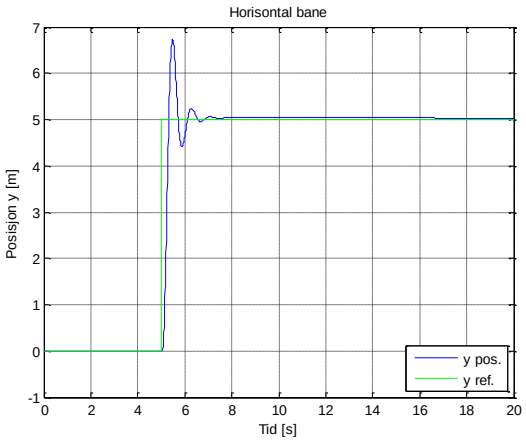
8.2.3 Full simulasjon med ulineær fly modell

Finner de ytre regulator parameterne

Siden vi har bruker en annen flydynamikk her enn i kapitel 4 må vi justere inn den ytre sløyfen på nytt med Zigler Nichlor's metode. Det viser seg at vi må etterjustere den vertikale regulatoren, og får følgende referanse forfølging med YTT.



Figur 80: Vertikal referanse forfølging

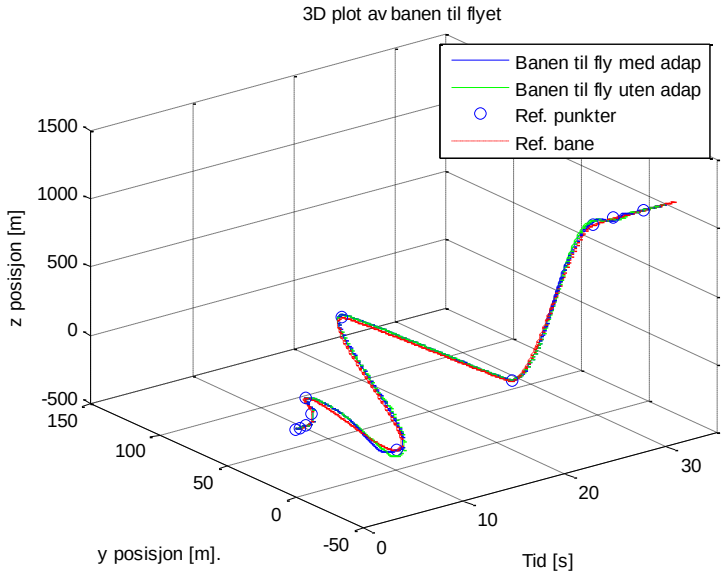


Figur 81: Horisontal referanse forfølging

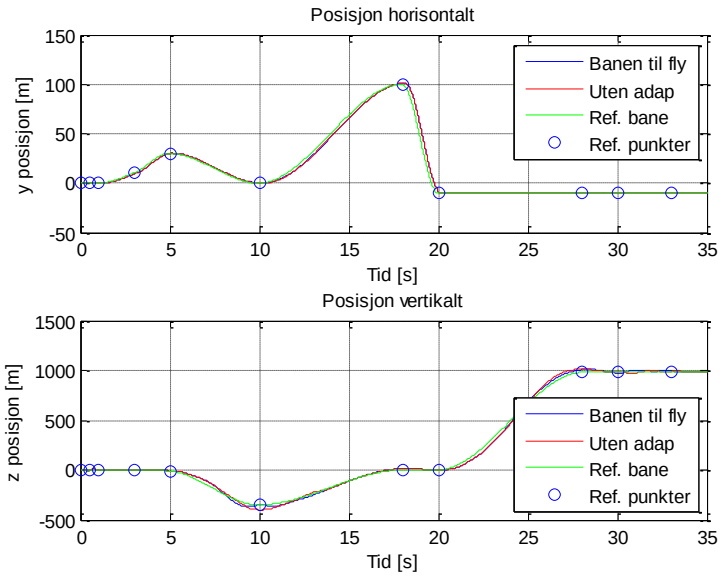
Her har vi passet på at angrepsvinkel ikke kommer over/under $\pm 8^0$.

Simulerer full flymodell med adaptiv regulator på pitch sløyfen

Vi simulerer full flymodell med BTT der vi legger inn punkter i rommet og banen blir regnet ut med Hermitian interpolasjon og et maks rorutslag på $\pm 25^0$. Vi tillater et stort rorutslag for lettere å få en angrepsvinkel over/under $\pm 8^0$.

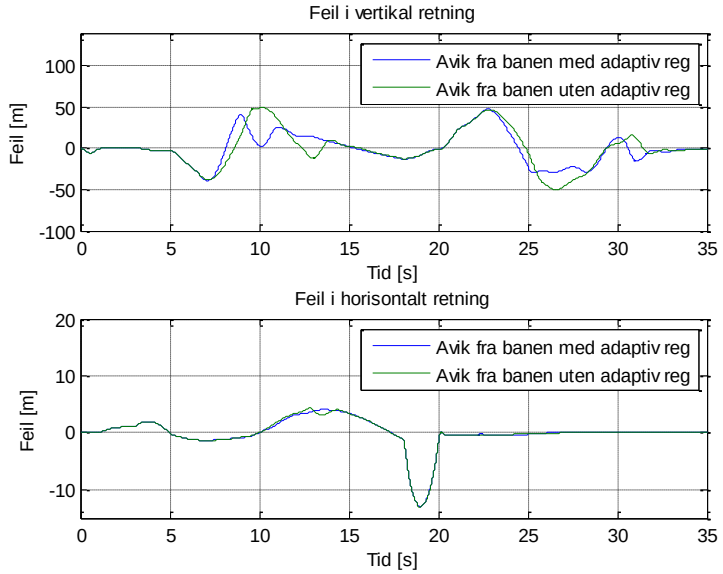


Figur 82: 3D plot av banen til flyet



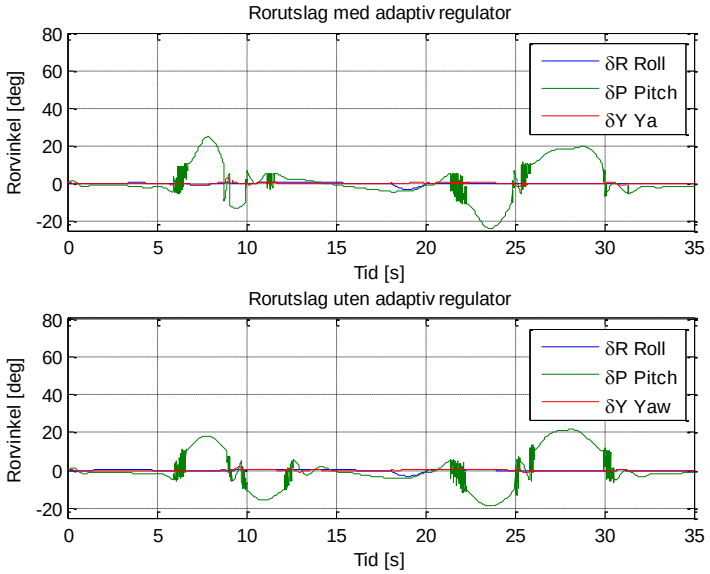
Figur 83: 2D plot av banen til flyet

Som man ser på figuren over og figur 84 har vi en forbedring med den adaptive regulatoren når man har en angrepsvinkel som er over/under $\pm 8^0$.

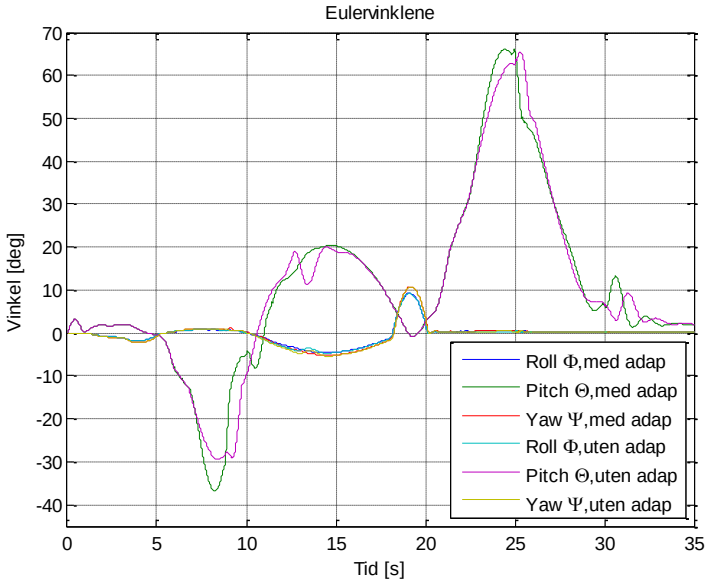


Figur 84: Avvik fra referansebanen

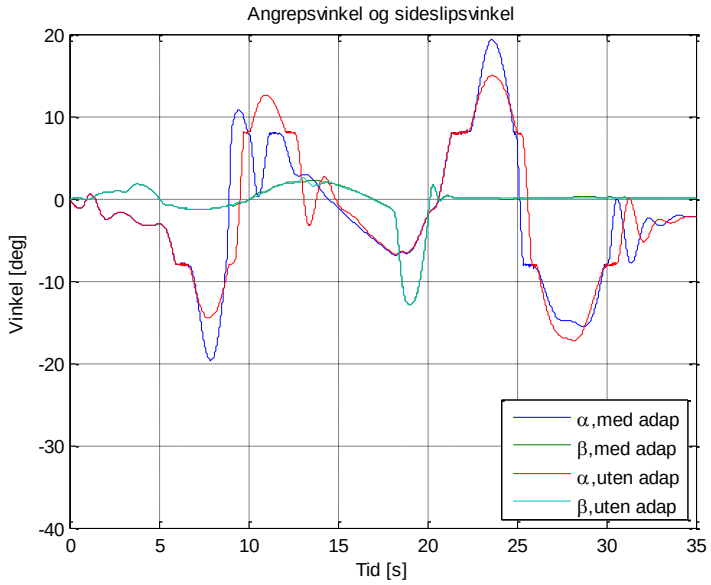
Ser at vi har null i statisk avvik for både vertikal og horisontal regulator.



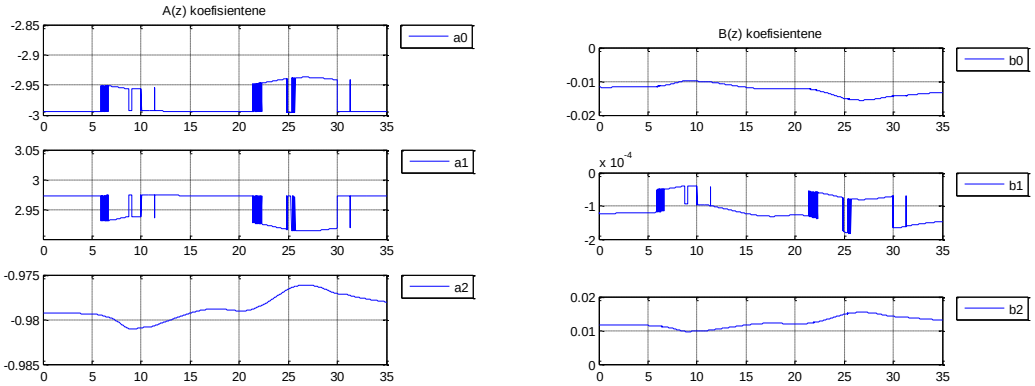
Figur 85: Rorutslag for adaptivt system og system uten adaptiv regulator



Figur 86: Eulervinklene til adaptiv regulator og uten adaptiv regulator



Figur 87: Angrepsvinkel og sideslipsvinkel



Figur 88: Det gjenkjente diskrete systemet, som er avhengig av $C_{M\alpha}$ og V_T

9 Konklusjon og videre arbeid

9.1 Konklusjon

I denne oppgaven har jeg vurdert forskjellige metoder for å implementere en adaptiv regulator på pitch-systemet til en UVA. Jeg har bruket Rekursive Least Sqare metoder og diskretisering av forenklet pitch-system til å identifisere systemet. For å regulere systemet har jeg brukt poltildelingsmetoden med varierende regulatorpoler.

Det har vist seg at de rekursive metodene ble ustabile når jeg hadde en forandring i pitch-vinkelen samtidig som det var en forandring av den aerodynamiske momentkoeffisienten. Hvis pitch-vinkelen derimot var konstant når skifte fant sted gikk det bra. Jeg så også at man greide å følge forandringen i systemet med varierende total hastighet. Det viste at man kunne følge små forandringer i systemet med Rekursive Least Sqare metoder. Siden jeg skulle se om man greide å gjenkjenne systemet når flyet fikk turbulens rundt vingene så ville ikke Rekursive Least Sqare metodene fungere. Et annet problem med disse metodene er at de krevde lik amplitude i referansesignalet som når den ble innstilt.

Når jeg brukte diskretisering av forenklet pitch-system så greide jeg å få systemet stabilt. For å få bedre respons med den adaptive regulatoren, brukte jeg forskjellige regulatorpoler for når det er et ustabilt og stabilt system. Da viste det seg at man får bedre respons jo lenger man holder seg i det stabile systemet, det vil si at angrepsvinkel er over/under $\pm 8^0$. Det er derfor anbefalt å bruke en adaptiv regulator som baserer seg på å diskretisere pitch-systemet, der man bruker tabelloppslag for å finne de aerodynamiske koeffisientene.

9.2 Videre arbeid

I det videre arbeidet vil jeg anbefale å bruke tabelloppslag for de aerodynamiske dataene som er oppdrevet fra vindtunnel målinger. På denne måten kan man se om den adaptive regulatoren har noen hensikt i det enkelte tilfelle. Deretter er det hensiktsmessig å implementere adaptive regulatorer for roll- og yaw-systemene med diskretisering av forenklet system i de respektive systemer.

Referanser

Fossen, Thor I. 2002. *Marine Control Systems.* s.l. : Marine Cybernetics AS, 2002. ISBN: 82-92356-00-2.

—. 1998. *Matematiske modeller for styring av fly og satelitter.* s.l. : Institutt for teknisk kybernetikk norges teknisk-naturvitenskapelige universitet, 1998. Rapportnummer 98-4-W.

Hallingstad, Oddvar. 2008. *Matematisk modellering av dynamiske systemer.* s.l. : UNIK notat, 2008.

Haugen, Finn. 2003a. *Dynamiske systemer.* 2. s.l. : Tapir akademisk forlag, 2003a. ISBN: 82-519-1877-4.

—. 2003b. *Praktisk reguleringsteknikk.* 2. s.l. : Tapir akademisk forlag, 2003b. ISBN: 82-519-1887-1.

—. 1996. *Regulering av dynamiske systemer.* s.l. : Tapir Forlag, 1996. ISBN: 82-519-1407-8.

McLean, Donald. 1990. *Automatic Fligth Control Systems.* s.l. : Prentice Hall, 1990. ISBN0-13-054008-0.

Skullestad, Åge. 2003. *Missile Dynamic 05.* s.l. : Upublisert manuskript, 2003.

—. 2007b. *Parametet Identification and Mathematical Modelling.* s.l. : Upublisert manuskript, 2007b.

—. 2007a. *Storkastiske systemer.* s.l. : Upublisert manuskript, 2007a.

Stivens, Brian L and Lewis, Frank L. 1992. *Aircraft control and simulation.* s.l. : John Wiley & Sons, Inc., 1992. pp. 1-111. ISBN: 0-471-61397-5.

Wellstead, P E and Zarrop, M B. 1991. *Self-tuning Systems Controll and Signal Processing.* s.l. : John Wiley & Sons Inc., 1991. ISBN: 0-471-93054-7.

Vedlegg

Vedlegg A: Teori

A.1 Linearisering

Vi har gitt en tilstandsrommodell

$$\underline{\dot{x}} = \underline{f}(\underline{x}, \underline{u}) \tag{A.1}$$

Vi antar at systemet befinner seg i arbeidspunktet

$$\underline{\dot{x}}_0 = \underline{f}(\underline{x}_0, \underline{u}_0) \tag{A.2}$$

Antar at $\underline{u}$ får et tilegg $\Delta\underline{u}$ fra $\underline{u}_0$ og $\underline{x}$ dermed et tillegg $\Delta\underline{x}$ fra $\underline{x}_0$. Da kan (A.2) skrives som

$$\frac{d(\underline{x}_0 + \Delta\underline{x})}{dt} = \underline{f}(\underline{x}_0 + \Delta x, \underline{u}_0 + \Delta\underline{u}) \tag{A.3}$$

$$\downarrow$$

$$\underline{\dot{x}}_o + \Delta\underline{\dot{x}} \approx \underline{f}(\underline{x}_0, \underline{u}_0) + \frac{\partial \underline{f}}{\partial \underline{x}}\bigg|_{\underline{x}_0, \underline{u}_0} \Delta\underline{x} + \frac{\partial \underline{f}}{\partial \underline{u}}\bigg|_{\underline{x}_0, \underline{u}_0} \Delta\underline{u} \tag{A.4}$$

Her har vi brukt 1. ordens Taylerrekkeutvikling av $f(.)$. Utnytter at $\underline{\dot{x}}_0$ er lik $\underline{f}(\underline{x}_0, \underline{u}_0)$, og da kan vi skrive (A.4) slik

$$\Delta\underline{\dot{x}} = \underbrace{\frac{\partial \underline{f}}{\partial \underline{x}}\bigg|_{\underline{x}_0, \underline{u}_0}}_{=A} \Delta\underline{x} + \underbrace{\frac{\partial \underline{f}}{\partial \underline{u}}\bigg|_{\underline{x}_0, \underline{u}_0}}_{=B} \Delta\underline{\mathrm{u}} = A\Delta\underline{\mathrm{x}} + \mathrm{B}\Delta\underline{\mathrm{u}} \tag{A.5}$$

På detaljertform får vi

$$\begin{bmatrix} \Delta\dot{x}_1 \\ \Delta\dot{x}_2 \\ \vdots \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}\Bigg|_{\underline{x}_0, \underline{u}_0}}_{=A} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \cdots \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}\Bigg|_{\underline{x}_0, \underline{u}_0}}_{=B} \begin{bmatrix} \Delta u_1 \\ \Delta u_2 \\ \vdots \end{bmatrix} \tag{A.6}$$

Som er den lokale lineære modellen. A og B blir her Jacobimatriser som generelt er funksjoner av arbeidspunktet. Derfor vil vi bruke MatLab funksjonen $jacobian()$ til å beregne disse matrisene.

Lineariseringen av målelikningen

$$\underline{y} = \underline{g}(\underline{x}, \underline{u}) \tag{A.7}$$

Er tilsvarende

$$\Delta \underline{y} = \underbrace{\frac{\partial \underline{g}}{\partial \underline{x}}\bigg|_{\underline{x}_0,\underline{u}_0}}_{=C} \Delta \underline{x} + \underbrace{\frac{\partial \underline{g}}{\partial \underline{u}}\bigg|_{\underline{x}_0,\underline{u}_0}}_{=D} \Delta \underline{u} = C\Delta \underline{x} + D\Delta \underline{u} \tag{A.8}$$

A.2 *adj* og *det*

adj betyr adjungert, der den adjungerte av A er den transpnerte av den matrisen vi får når hvert ellement a_{ij} i A erstattes med sin kofaktor C_{ij}. Kompakt uttrykt får vi

$$adj(A) = \left(C_{ij}\right)^T \tag{A.9}$$

det(A) er determinanten til en matrise A.

A.3 Hvordan komme frem til diskret transferfunksjon fra tilstandsrommodeller

Er gitt tilstandsrommodellen

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{A.10}$$

Der det går å finne transferfunksjonen fra u til y. Vi begynner med å ta Laplacetransformasjonen av likning (A.10) og får

$$\begin{aligned} sx(s) - x_0 &= Ax(s) + Bu(s) \\ y(s) &= Cx(s) + Du(s) \end{aligned} \tag{A.11}$$

Der x_0 er initialtilstanden. Fra (A.11) får vi

$$(sI - A)x(s) = x_o + Bu(s) \tag{A.12}$$

Premultiplisering med $(sI - A)^{-1}$ på begge sider

$$x(s) = (sI - A)^{-1}x_0 + (sI - A)^{-1}Bu(s) \tag{A.13}$$

Kombinerer vi (A.11) med (A.13) får vi

$$y(s) = C(sI - A)^{-1}\, x_0 + (C(sI - A)^{-1}B + D)u(s) \tag{A.14}$$

Transferfunksjonen fra u til y er da

$$H(s) = \frac{y(s)}{u(s)} = C(sI - A)^{-1}B + D \tag{A.15}$$

Der $(sI - A)^{-1}$ kan finnes fra

$$(sI - A)^{-1} = \frac{\mathrm{adj}(sI - A)}{\det(sI - A)} \tag{A.16}$$

Vedlegg B: Flydata

B.1 Parametere som inngår i modellen

De aerodynamiske koeffisientene er linearisert ved en $V_T = 289\ [^m/_s]$. De aerodynamiske koeffisientene som ikke er nevnt under er null.

Flyet:

$m = 325{,}7[kg]$
$I_{xx} = 7{,}3243\ [Nm]$
$I_{yy} = 177{,}3608\ [Nm]$
$I_{zz} = 177{,}9664\ [Nm]$
$I_{xz} = 1\ [Nm]$

Areal:

$S = 0{,}75\ [m^2]$
$b = 1{,}3\ [m^2]$
$c = 0{,}47\ [m^2]$

Generelle konstanter:

$g = 9{,}81\ \left[^m/_{s^2}\right]$
$\rho = 1.2928\ \left[\frac{N}{m^2}\right]$

Aerodynamiske koeffisienter:

Dreg:

$C_{D0} = 0{,}1719$
$C_{D\alpha} = 0.0057\ \left[^1/_{rad}\right]$

Lift:

$C_{L\alpha} = 3{,}8969\ \left[^1/_{rad}\right]$
$C_{L\delta_1} = 0{,}1998\ \left[^1/_{rad}\right]$
$C_{L\delta_2} = 0{,}1998\ \left[^1/_{rad}\right]$
$C_{L\delta_3} = 0{,}1998\ \left[^1/_{rad}\right]$
$C_{L\delta_4} = 0{,}1998\ \left[^1/_{rad}\right]$

Sidekraft:

$C_{C\beta} = 0{,}8305\ \left[^1/_{rad}\right]$
$C_{C\delta_1} = -0{,}1677\ \left[^1/_{rad}\right]$
$C_{C\delta_2} = 0{,}1677\ \left[^1/_{rad}\right]$
$C_{C\delta_3} = 0{,}1677\ \left[^1/_{rad}\right]$
$C_{C\delta_4} = -0{,}1677\ \left[^1/_{rad}\right]$

Roll moment:

$C_{\bar{L}\beta} = -0{,}1411\ \left[^1/_{rad}\right]$
$C_{\bar{L}P} = -1{,}4567\ \left[^1/_{rad}\right]$
$C_{\bar{L}R} = -1{,}1827\ \left[^1/_{rad}\right]$
$C_{\bar{L}\delta_1} = 0{,}1394\ \left[^1/_{rad}\right]$
$C_{\bar{L}\delta_2} = -0{,}1394\ \left[^1/_{rad}\right]$
$C_{\bar{L}\delta_3} = 0{,}1603\ \left[^1/_{rad}\right]$
$C_{\bar{L}\delta_4} = -0{,}1603\ \left[^1/_{rad}\right]$

Pitch moment:

$C_{M\alpha} = -0{,}3951\left[^1/_{rad}\right]$
$C_{MQ} = -15{,}4152\left[^1/_{rad}\right]$
$C_{M\delta_1} = -0{,}4507\left[^1/_{rad}\right]$
$C_{M\delta_2} = -0{,}4507\left[^1/_{rad}\right]$
$C_{M\delta_3} = -0{,}4507\left[^1/_{rad}\right]$
$C_{M\delta_4} = -0{,}4507\left[^1/_{rad}\right]$

Yaw moment:

$C_{N\beta} = 0{,}0457\left[^1/_{rad}\right]$
$C_{NP} = -5{,}6989\left[^1/_{rad}\right]$
$C_{NR} = -7{,}9635\left[^1/_{rad}\right]$
$C_{N\delta_1} = -0{,}3781\left[^1/_{rad}\right]$
$C_{N\delta_2} = 0{,}3781\left[^1/_{rad}\right]$
$C_{N\delta_3} = 0{,}3781\left[^1/_{rad}\right]$
$C_{N\delta_4} = -0{,}3781\left[^1/_{rad}\right]$

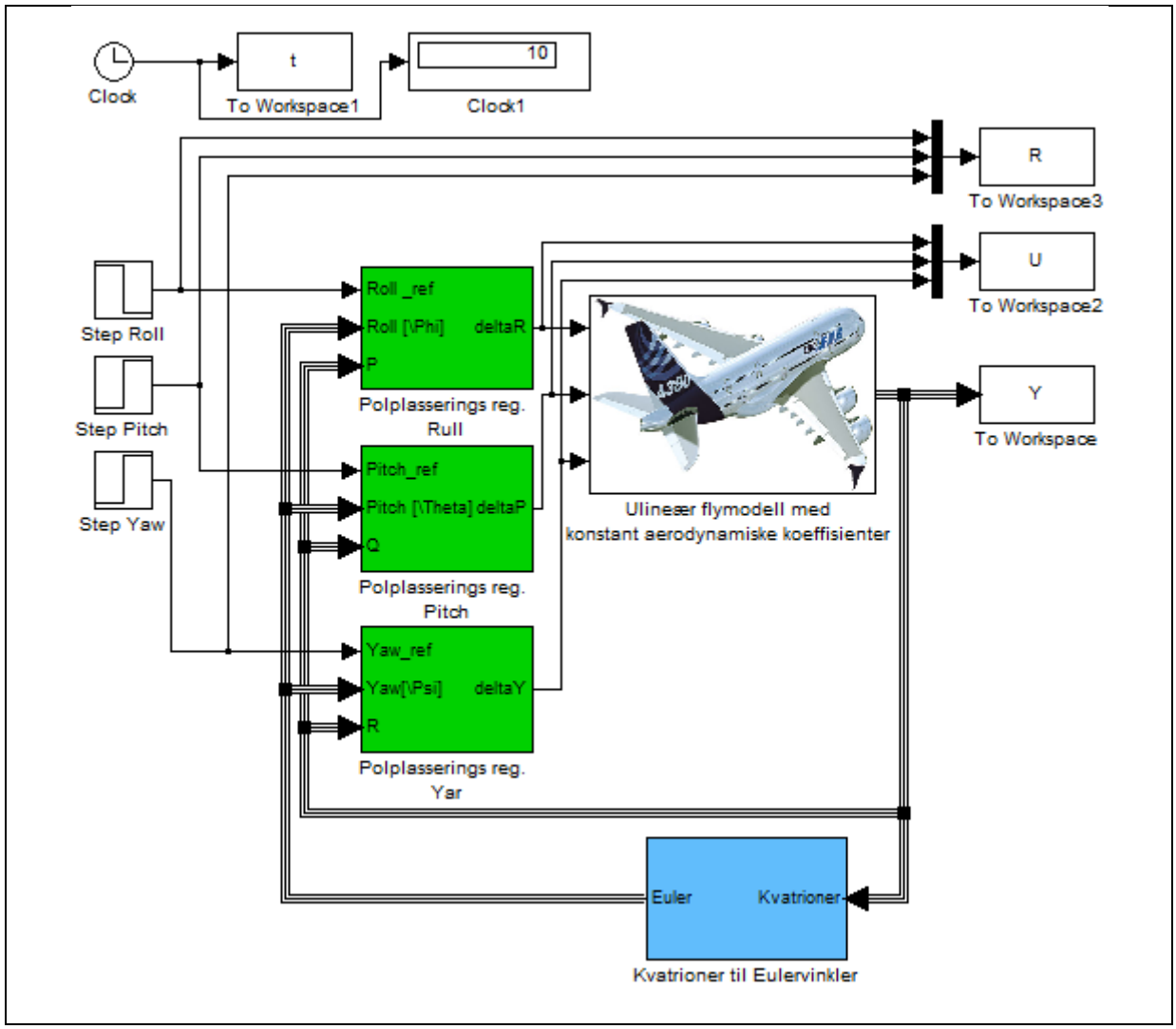
Vedlegg C: Matlab kode og simulink modeller

Skript og funksjoner som er felles for flere simulasjoner er samlet til slutt. Simulink blokker som er farget er sub- systemer og er beskrevet i vedlegg C.5.

C.1 Kode kapitel 4 -Utvikling av indre og ytre regulatorer

C.1.1 Indre sløyfe

C.1.1.1 Simulink modell



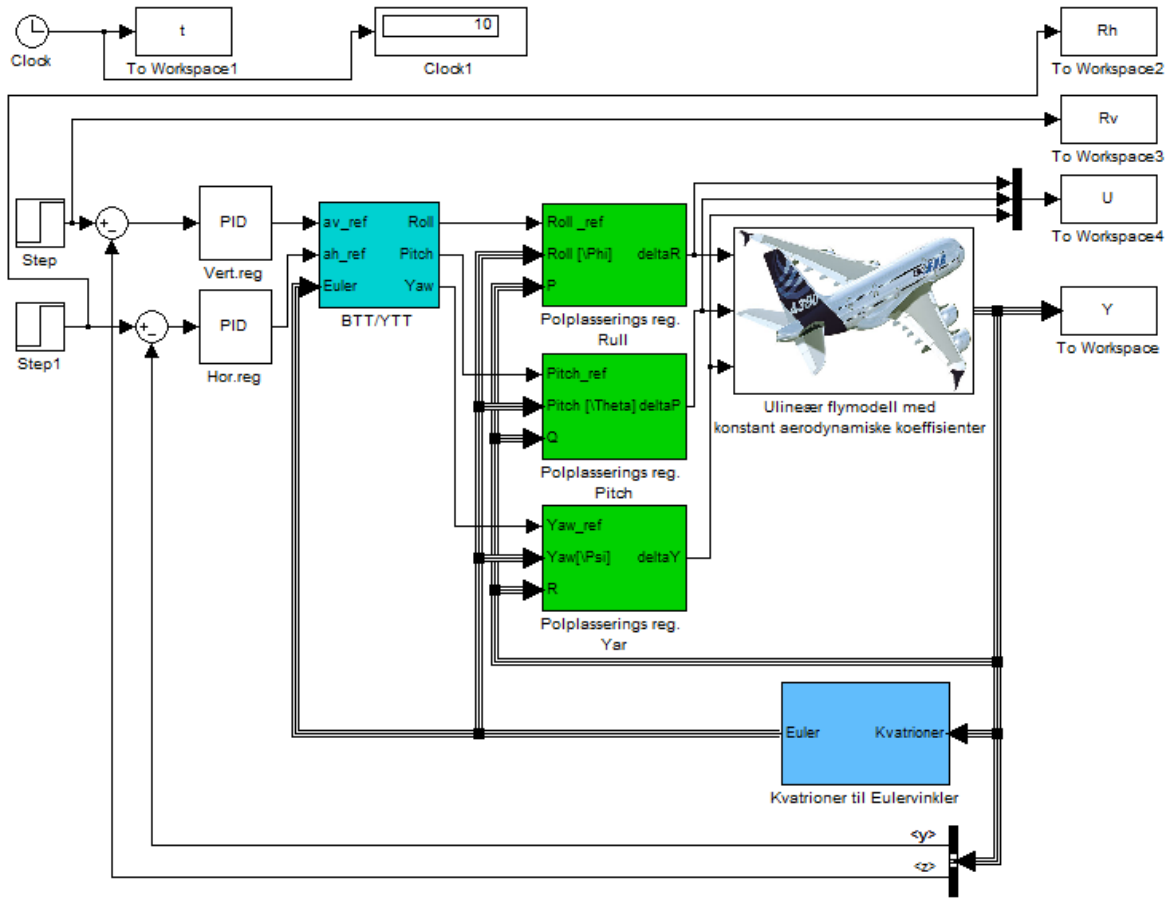
C.1.1.2 Skript fil

```
%-------Masteroppgave -------------
%Dato: 18.05.10
%Av Thomas Algarheim
%   Simulering av fly med kvatrioner, indre sløyfe som følger
%   referansepunkter

clc
clear all
%% Initialisering
disp('Initialiserin')
Initialisering
```

```
global Ts
Ts=0.01; %Sampel tiden
Time=10; %Lengde på simulasjon

%Startbetingelse
 x0=[200,0*g2r,   0, 1, 0, 0, 0,0,0,0,0,0,0,200,0,0,]';
%x0=[V_T,alpha,beta,q0,q1,q2,q3,P,Q,R,x,y,z,  U,V,W,]';

%Lineariserer for indre sløyfe
disp('Lineariserer de tre indre sløyfene')
Linearisering_av_indre_loop

%Setter opp regulatorene
disp('Setter opp regulatorene')
[Gr Gp Gy]=IndreReg(Ar, Br, Ap, Bp, Ay, By, 1,1);

% Kjører system
disp('Kjører systemet...')
open_system('Indre_loop_ulinmod_pol_sim')
[t,X]=sim('Indre_loop_ulinmod_pol_sim');
% Skriver ut resultater
disp('Skriver ut resultatet')
figure(1)
subplot(2,1,1);
plot(t,R(:,1).*r2g,'b',t,R(:,2).*r2g,'r',t,R(:,3).*r2g,'g',...
    t,Euler(1,:).*r2g,'b',t,Euler(2,:).*r2g,'r',t,Euler(3,:).*r2g,'g')
title('Utskrift av posisjon til fly i forhold til referanse posisjonen')
legend('Roll \Phi ref','Pitch \Theta ref', 'Yaw \Psi ref',...
       'Roll \Phi','Pitch \Theta', 'Yaw \Psi',-1)
xlabel('Tid [s]')
ylabel('Vinkel [deg]')
subplot(2,1,2);
plot(t,U(:,1).*r2g,'b',t,U(:,2).*r2g,'r',t,U(:,3).*r2g,'g')
title('Utskrift av rorutslag')
legend('\deltaR','\deltaP','\deltaY',-1)
xlabel('Tid [s]')
ylabel('Vinkel [deg]')
```

C.1.2 Ytre sløyfe med YTT/BTT

C.1.2.1 Simulinkmodell



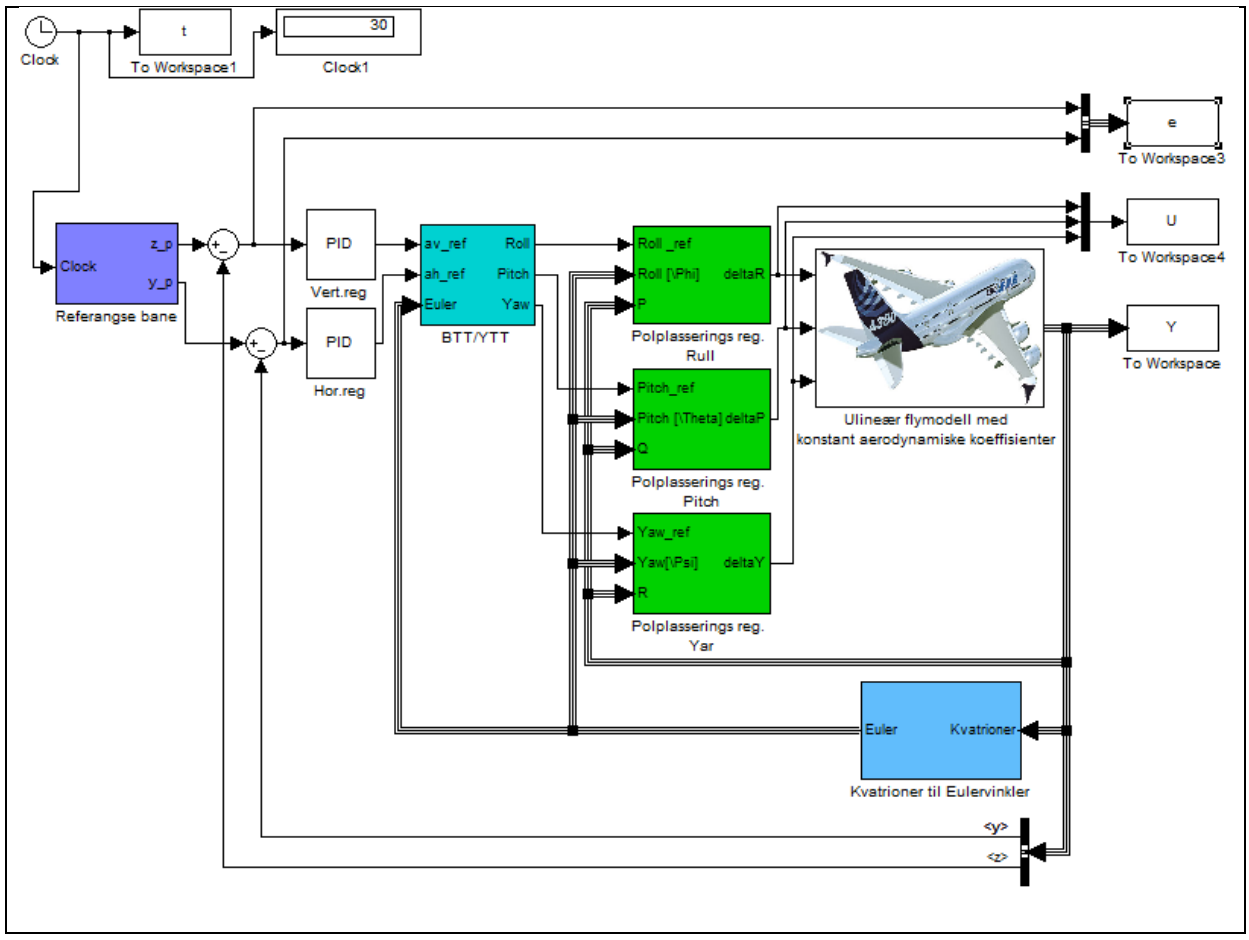
C.1.2.2 Skript fil

```
%-------Masteroppgave -------------
%Dato: 19.05.10
%Av Thomas Algarheim
%   Simulering av fly med kvatrioner, som følger et step i ytre sløyfe

clc
clear all
%% Initialisering
disp('Initialiserin')
Initialisering

global Ts
Ts=0.01; %Sampel tiden
Time=10; %Lengde på simulasjon

%Startbetingelse
 x0=[200,0*g2r,    0, 1, 0, 0, 0,0,0,0,0,0,0,200,0,0,]';
%x0=[V_T,alpha,beta,q0,q1,q2,q3,P,Q,R,x,y,z,  U,V,W,]'

%Lineariserer for indre sløyfe
disp('Lineariserer de tre indre sløyfene')
Linearisering_av_indre_loop
```

```
%Setter opp regulatorene
disp('Setter opp regulatorene')

[Gr Gp Gy]=IndreReg(Ar, Br, Ap, Bp, Ay, By, 1,1);

%Regulator ytre sløyfe
Kpk_ver=0.047;              %Kritisk forsterkning vertikal reg.
Tp_ver=(5.85-5.3)/60;       %Kritisk periode vertikal reg.
[Kp_ver Ti_ver Td_ver Tf_ver]=YtreReg(Kpk_ver,Tp_ver,2);

Kpk_hor=0.067;              %Kritisk forsterkning horisontal reg.
Tp_hor=(5.58-5.22)/60;      %Kritisk periode horisontal reg.
[Kp_hor Ti_hor Td_hor Tf_hor]=YtreReg(-Kpk_hor,-Tp_hor,2);

%Bank-To-Turn (BBT)/ Yaw-To-Trun (YTT)
% btt=1: BTT
% btt=0: YTT

btt=1;
Kbtt1=5;
Kbtt2=1;
Kbtt3=0.1;

%Kjører system
step_vert=0;
step_hor=5;
disp('Kjører systemet...')
open_system('Ytre_loop_med_YTT_BTT_sim')
[t,X]=sim('Ytre_loop_med_YTT_BTT_sim');

%Skriver ut resultater
disp('Skriver ut resultatet')

figure(1)
plot(t,Euler(1,:).*(180/pi),t,Euler(2,:).*(180/pi),t,Euler(3,:).*(180/pi))
title('Stillingen til flyet')
legend('Roll \Phi','Pitch \Theta', 'Yaw \Psi',4)
xlabel('Tid [s]')
ylabel('Vinkel [deg]')
grid on

figure(2)
plot(t,Y(:,12),'b',t,Rh(:,1),'g'); grid on
title('Horisontal bane')
ylabel('Posisjon y [m]'), xlabel('Tid [s]');
legend('y pos.','y ref.',4)

figure(3)
plot(t,Y(:,13),'b',t,Rv(:,1),'g');grid on
title('Vertikal bane')
ylabel('Posisjon z [m]'), xlabel('Tid [s]');
legend('z pos.','z ref.',4)

%%
disp('Slutt')
```

C.1.3 Simulerer fullt system ut i fra referansepunkter

C.1.3.1 Simulinkmodell



C.1.3.2 Skript fil

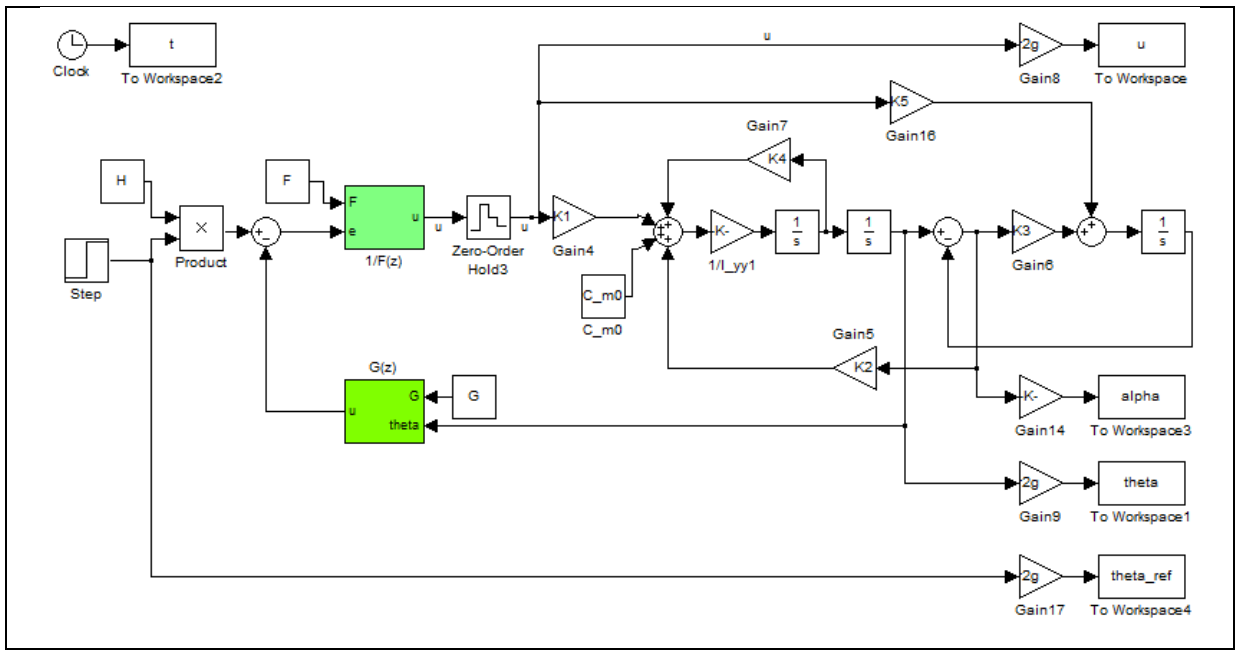
```
%-------Masteroppgave -------------
%Dato: 19.05.10
%Av Thomas Algarheim
%   Simulering av fly med kvatrioner, som følger referansepunkter

clc
clear all
%Initialisering
disp('Initialiserin')
Initialisering

global Ts
Ts=0.01; %Sampel tiden
Time=30; %Lengde på simulasjon

%Startbetingelse
 x0=[200,0*g2r,    0, 1, 0, 0, 0,0,0,0,0,0,0,200,0,0,]';
%x0=[V_T,alpha,beta,q0,q1,q2,q3,P,Q,R,x,y,z,  U,V,W,]'

%Setter opp bane til flyet
disp('Finner bane ut i fra referangse punkter')
wpt_x = [0 100 200 600  1000  2000  3600    4000]; %ca 200 m/s
wpt_y = [0 0    0   10    30     0     -10       -10];
```

```
wpt_z = [0 0    5   5   -10   -350     20       20];
wpt_t = [0 0.5  1   3    5     10    18      20];
[x_p y_p z_p tbane] = Bane(wpt_x, wpt_y, wpt_z, wpt_t, Ts, Time+1);

%Lineariserer for indre sløyfe
disp('Lineariserer de tre indre sløyfene')
Linearisering_av_indre_loop

%Setter opp regulatorene
disp('Setter opp regulatorene')

%Regulator indre sløyfe
[Gr Gp Gy]=IndreReg(Ar, Br, Ap, Bp, Ay, By, 1,1);

%Regulator ytre sløyfe
Kpk_ver=0.047;              %Kritisk forsterkning vertikal reg.
Tp_ver=(5.85-5.3)/60;       %Kritisk periode vertikal reg.
[Kp_ver Ti_ver Td_ver Tf_ver]=YtreReg(Kpk_ver,Tp_ver,2);

Kpk_hor=0.067;              %Kritisk forsterkning horisontal reg.
Tp_hor=(5.58-5.22)/60;      %Kritisk periode horisontal reg.
[Kp_hor Ti_hor Td_hor Tf_hor]=YtreReg(-Kpk_hor,-Tp_hor,2);

%Bank-To-Turn (BBT)/ Yaw-To-Trun (YTT)
% btt=1: BTT
% btt=0: YTT

btt=1;
Kbtt1=5;
Kbtt2=1;
Kbtt3=0.1;

%Kjører system
disp('Kjører systemet...')
open_system('Full_mod_stab_fly_bane_sim')
[t,X]=sim('Full_mod_stab_fly_bane_sim');

%Skriver ut resultater
figure(1)
plot(t,Y(:,8)*180/pi,'b',t,Y(:,9)*180/pi,'r',t,Y(:,10)*180/pi,'g')
title('Vinkelhastigherer')
legend('P','Q','R',4)
xlabel('Tid [s]')
ylabel('Vinkelhastighet[grader/s]')
grid on

figure(2)
plot(t,Y(:,2).*(180/pi),'b',t,Y(:,3).*(180/pi),'r')
title('Angrepsvinkel og sideslipsvinkel')
legend('\alpha','\beta',4)
xlabel('Tid [s]')
ylabel('Vinkel [grader]')
grid on

figure(3)
plot(t,Euler(1,:).*(180/pi),t,Euler(2,:).*(180/pi),t,Euler(3,:).*(180/pi))
title('Eulervinklene')
legend('Roll \Phi','Pitch \Theta', 'Yaw \Psi',4)
xlabel('Tid [s]')
```

```
ylabel('Vinkel [grader]')
grid on

figure(4)
plot(t,U(:,1).*r2g,'b',t,U(:,2).*r2g,'g',t,U(:,3).*r2g,'r')
title('Rorutslag ')
ylabel('Rorvinkel [deg]'), xlabel('Tid [s]');
legend('\deltaR Roll','\deltaP Pitch','\deltaY Yaw',4)
grid on

figure(5)
subplot(2,1,1);plot(t,Y(:,12),'b',tbane,y_p,'g',wpt_t,wpt_y,'o');grid on
title('Posisjon horisontalt')
ylabel('Posisjon y'), xlabel('Tid [s]');
legend('Banen til fly','Ref. bane','Ref. punkter',1)
subplot(2,1,2);plot(t,Y(:,13),'b',tbane,z_p,'g',wpt_t,wpt_z,'o')
title('Posisjon vertikalt')
ylabel('Posisjon z'), xlabel('Tid [s]');
legend('Banen til fly','Ref. bane','Ref. punkter',4)
grid on

figure(6)
plot3(t,Y(:,12),Y(:,13),'b',wpt_t,wpt_y,wpt_z,'o',tbane,y_p,z_p,'r-.')
grid on
title('3D plot av banen til flyet')
legend('Banen til fly','Ref. punkter','Ref. bane',1)
xlabel('tid')
ylabel('y pos.')
zlabel('z pos.')

figure(7)
subplot(2,1,1);plot(t,e(:,1),'b');grid on
title('Feil i vertikal retning')
ylabel('Feil [m]'); xlabel('Tid [s]')
legend('Feil',4)
subplot(2,1,2);plot(t,e(:,2),'b');grid on
title('Feil i horisontalt retning')
ylabel('Feil [m]'); xlabel('Tid [s]')
legend('Feil',4)
```

C.2 Kode kapitel 6 - Regulerer pitch-systemet med forskjellige $C_{M\alpha}$

C.2.1.1 Simulinkmodell



C.2.1.2 Skript fil

```
%-------Masteroppgave -------------
%Dato: 19.05.10
%Av Thomas Algarheim
%   Simulering av forenklet pitch-system med Poltildelingskontroller

clc
clear all
%% Initialisering
disp('Initialiserin')
Initialisering %i=2 variable aerodynamikk koeffisient C_M

Ts=0.01; %Sampel tiden
Time=10; %Lengde på simulasjon

disp('Finner diskret transferfunksjon for pitch')
V_T=200;
C_malpha=-0.1*1/g2r;

[Az,Bz]=Dtf(V_T,C_malpha, Ts);
% tf(Bz',Az',Ts)

K1=0.5*rho*S*V_T^2*c*C_mdeltaP;
K2=0.5*rho*S*V_T^2*c*C_malpha;
K3=1/(2*m)*rho*S*V_T*(C_Lalpha);
K4=0.25*rho*S*V_T*c^2*C_mQ;
K5=1/(2*m)*rho*S*V_T*(C_LdeltaP);

%% Poler og nullpunkter til det diskrete systemet
disp('Finner polene og nullpunktene til det diskrete systemet')
Hd=dfilt.df1(Bz,Az);
zplane(Hd)
title({'Poler (x) og nullpunkter (o) for H(z^{-1})' 'C_{M\alpha}=' ...
```

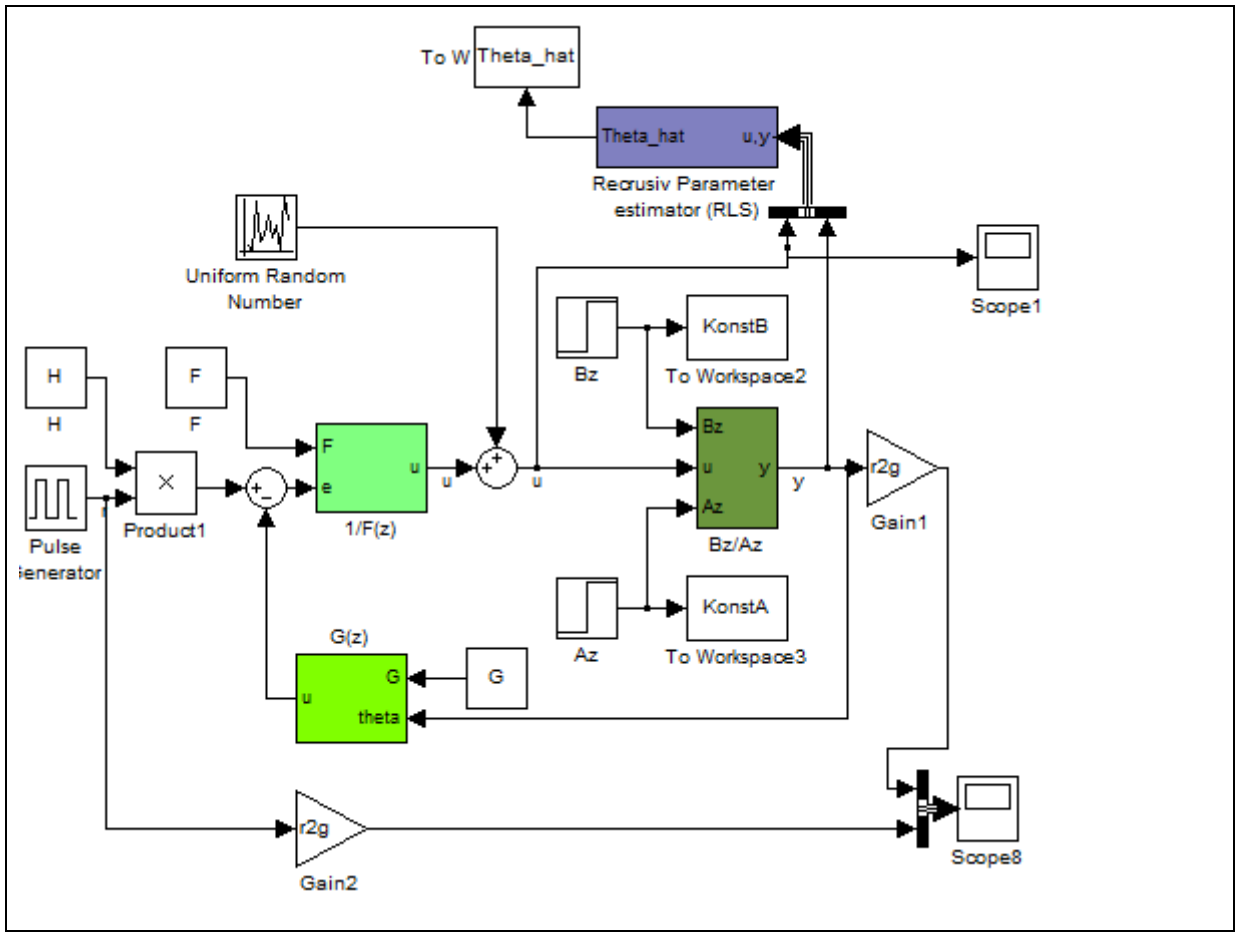
```
    C_malpha*(1/r2g)})
pz=roots(Az)
%% Finner lukket sløyfe poler
disp('Finner lukket sløyfe poler')
%Velger mellom sammenfallende poler eller Butterworth polynom
i=1;
if i==1
   %Sammenfallende poler
    w0=2.8; %Knekkfrekvensen
    syms sp
    n=2;
    ar1 = (sp+w0)^n;
    ar2=sym2poly(ar1);
    ar=conv(ar2,[1 w0]);
    regpol=roots(ar);
else
    %Butterworth polynom
    wr0=2.8; %Knekkfrekvensen
    n=3;
    alle_poler=roots([(-1)^n, zeros(1,2*n-1),1]);
    regpol=alle_poler(find(real(alle_poler)<0))*wr0;
end
syms s
[Bd,Ad]=tfdata(c2d(tf(1,(sym2poly((s-regpol(1))*(s-regpol(2))*...
    (s-regpol(3))))),Ts),'v');
dregpol=roots(Ad);

syms sp
Tp=sym2poly((sp-(dregpol(1)))*(sp-(dregpol(2)))*(sp-dregpol(3)))%
% Tp=sym2poly((sp-(0.9724))*(sp-(0.9724))*(sp-(0.9724)));%Ts=0.01;

pr=roots(Tp)

%% Setter opp pitch regulatoren med poltildelingsmetoden
disp('Setter opp pitch regulatoren med poltildelingsmetoden ')
[F G H]=PitchReg(Az,Bz,Tp);

%% Kjører system
disp('Kjører systemet...')
open_system('Pol_tildelings_sim')
sim('Pol_tildelings_sim');clear tout

%% Skriver ut resultater
figure(2)
subplot(2,1,1);plot(t,theta,'r',t,theta_ref,'b'); grid
legend('\theta [grader]','\theta_{ref}',4)
title({'Stiling til flykropp i pitch der' 'C_{M\alpha}=' C_malpha*(1/r2g)})
xlabel('Tis [s]')
ylabel('Vinkel [greder]')

subplot(2,1,2);plot(t,u,'b'); grid
legend('\deltaP [deg]',4)
xlabel('Tid [s]')
ylabel('Vinkel [deg]')
title('Pådrag')

figure(3)
plot(t,alpha,'g'); grid
legend('\alpha [deg]',4)
xlabel('Tid [s]')
```

```
ylabel('Vinkel [deg]')
title('Angrepsvinkel')

disp('Slutt')
```

C.3 Kode kapitel 7 - Identifisering av flymodellen i pitch-systemet

C.3.1 Tester ut RLS algoritmene på lineær modell

C.3.1.1 Simulinkmodell



C.3.1.2 RLS Skript

```
%-------Masteroppgave -------------
%Dato: 19.05.10
%Av Thomas Algarheim
%   Tester ut RLS Algoritmenen

clc
clear all

%% Initialisering
disp('Initialiserin')
Initialisering %i=2 variable aerodynamikk koeffisient C_M

Ts=0.01; %Sampel tiden
Time=20; %Lengde på simulasjon

disp('Finner diskret transferfunksjon for pitch')
```

```
V_T=200;
[Az(:,1),Bz(:,1)]=Dtf(V_T,C_malpha, Ts);
C_malpha=-0.1*1/g2r;
[Az(:,2),Bz(:,2)]=Dtf(V_T,C_malpha, Ts);

na=length(Az)-1;
nb=length(Bz)-1;

%% Setter opp pitch regulatoren med poltildelingsmetoden
disp('Setter opp pitch regulatoren med poltildelingsmetoden ')
syms sp
Tp=sym2poly((sp-0.9802)*(sp-0.9802)*(sp-0.9802));%Ts=0.01;
[F G H]=PitchReg(Az,Bz,Tp);


%% RLS---start---
RLS=1;
Kr=zeros(6,1);
Pr=10000*eye(6);
R=diag([0 0 0 0 0 0]);
lambda=0;
wb=2;
Threshold=5;
Theta_hat_init=[Az(2,1), Az(3,1), Az(4,1), Bz(1,1), Bz(2,1), Bz(3,1)]';


%% Kjører RLS_sim
disp('Kjører systemet...')
open_system('RLS_sim')
[t,X]=sim('RLS_sim');clear tout

%Skriver ut resultater
disp('Skriver ut resultatene for RLS')
Plot_id(t,Theta_hat,KonstA,KonstB)

%% RLS Random Walk
RLS=2;
Pr=100000*eye(6);
R=diag([500000 500000 0 0.09 60 0.01]);

%Kjører Random Walk
disp('Kjører systemet...')
[t,X]=sim('RLS_sim');clear tout

%Skriver ut resultater
disp('Skriver ut resultatene for Random Walk')
Plot_id(t,Theta_hat,KonstA,KonstB)

%% RLS med Forgeting Factor
RLS=3;
Pr=1000*eye(6);
lambda=0.96;
% wb=2;
% Threshold=5;

%Kjører Forgeting Factor
disp('Kjører systemet...')
[t,X]=sim('RLS_sim');clear tout
```

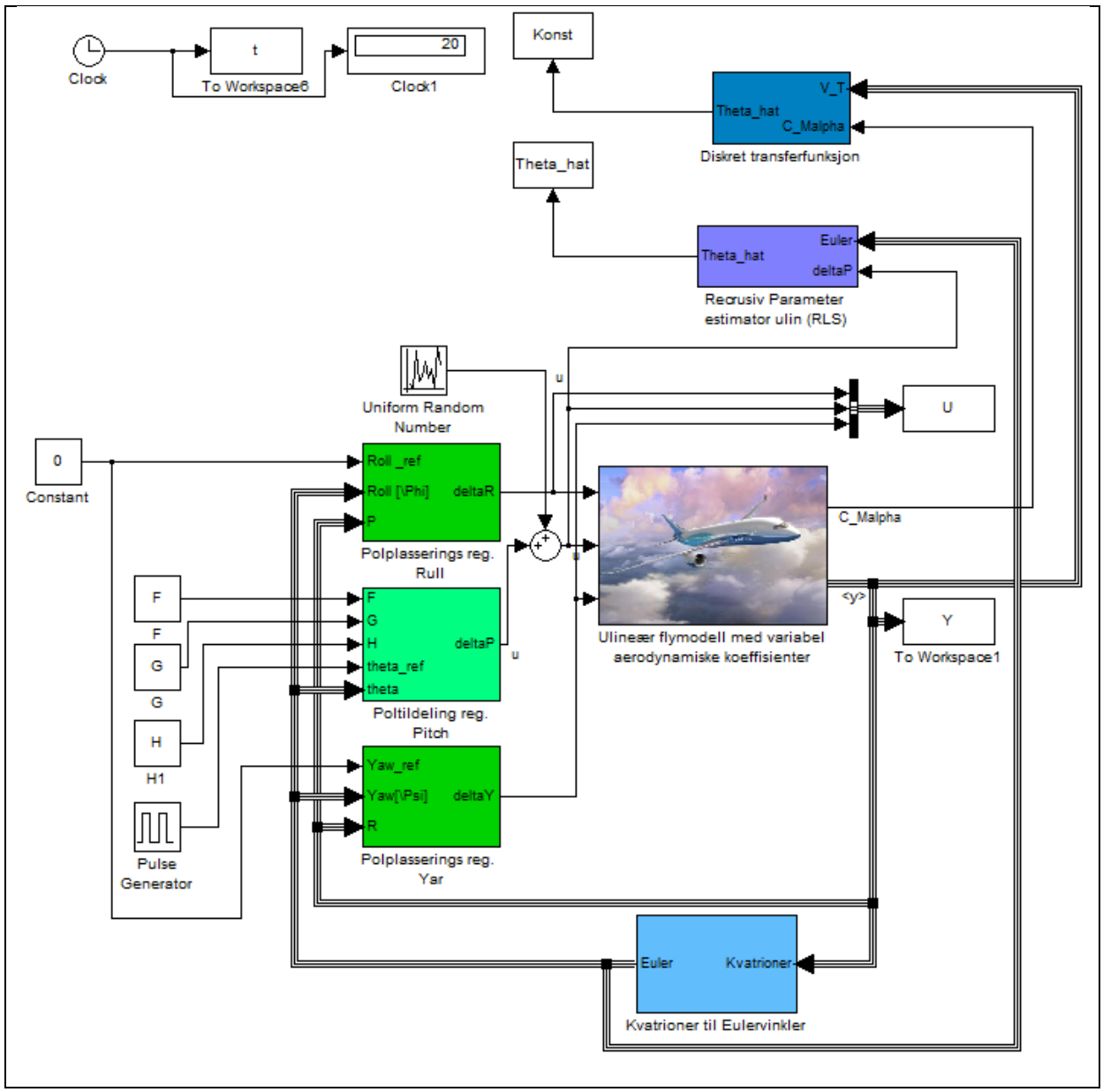
```
%Skriver ut resultater
disp('Skriver ut resultatene for Forgeting Factor')
Plot_id(t,Theta_hat,KonstA,KonstB)

%% RLS med konstant Trace
RLS=4;
Pr=100*eye(6);

%Kjører Konstant_trace
disp('Kjører systemet...')
[t,X]=sim('RLS_sim');clear tout

%Skriver ut resultater
disp('Skriver ut resultatene for konstant Trace')
Plot_id(t,Theta_hat,KonstA,KonstB)


disp('Slutt')
```

C.3.1.2 Plot for RLS skript fil

```
function Plot_id(t,Theta_hat,KonstA,KonstB)
%Dato: 19.05.10
%Av Thomas Algarheim
%   Plot for RLS Skript
na=3;
nb=2;
figure;cla; %Figur A(z)
colors = hsv(na); labels = {'data'};
hold on
for d = 1:(na) %plot b
    subplot(3,1,d);plot(t,KonstA(:,d+1)'...
        ,t,Theta_hat(:,d),'color',colors(d,:));
    if d==1
        title('A(z) koeffisientene')
    end
    labels{1} = ['a' int2str(d)];
    labels{2} = ['est.a' int2str(d)];
    legend(labels,-1);
 end
hold off

figure;cla; %Figur B(z)
colors = hsv(nb+1); labels = {'data'};
hold on
for d = 1:(nb+1) %plot b
    subplot(3,1,d);plot(t,KonstB(:,d)',...
        t,Theta_hat(:,d+na),'color',colors(d,:));
    if d==1
        title('B(z) koeffisientene')
    end
    labels{1} = ['b' int2str(d)-1];
    labels{2} = ['est.b' int2str(d)-1];
    legend(labels,-1);
end
hold off
```

C.3.2 Tester ut RLS algoritmene på ulineær modell med variabel $C_{M\alpha}$

C.3.2.1 Simulinkmodell



C.3.2.2 Ulin. RLS Skript

```
%-------Masteroppgave 1.0.1 -------------
%Dato: 20.05.10
%Av Thomas Algarheim
%   Simulering av det ulineære flyet med kvatrioner der vi identifiserer
%   flyet med RLS med Random Walk og RLS med Forgaeting Factor

clc
clear all
%% Initialisering
disp('Initialiserin')
Initialisering %i=2 variable aerodynamiskk koeffisient C_M

global Ts
Ts=0.01; %Sampel tiden
Time=20; %Lengde på simulasjon
```

```
%Startbetingelse
V_T=200;
 x0=[V_T,0*g2r,   0, 1, 0, 0, 0,0,0,0,0,0,0,200,0,0,]';
%x0=[V_T,alpha,beta,q0,q1,q2,q3,P,Q,R,x,y,z,  U,V,W,]'

%Lineariserer for indre sløyfe
disp('Lineariserer de tre indre sløyfene')
Linearisering_av_indre_loop

%Setter opp regulatorene
disp('Setter opp regulatorene')
[Gr Gp Gy]=IndreReg(Ar, Br, Ap, Bp, Ay, By, 1,2);

%Reg Pitch
[Az,Bz]=Dtf(V_T,C_malpha, Ts);
syms sp
Tp=sym2poly((sp-(0.98))*(sp-(0.98))*(sp-(0.8)));%Ts=0.01;
[F G H]=PitchReg(Az,Bz,Tp);

%% RLS Random Walk---start---
RLS=1;  %Velger Random Walk
lambda=0;
Kr=zeros(6,1);
Pr=10000*eye(6);
R=diag([100 100 10 0.09 0.01 0.09]);
wb=3;
Threshold=5;
Theta_hat_init=[Az(2), Az(3), Az(4), Bz(1), Bz(2), Bz(3)]';
%RLS---slutt----

%Kjører system
disp('Kjører systemet...')
open_system('ulin_RLS_sim')
t=sim('ulin_RLS_sim');

%Skriver ut resultater for Random Walk
disp('Skriver ut resultatet for Random Walk')
Plot_idu(t,Y,Euler,Konst,Theta_hat,r2g)

%% RLS Forgeting Factor---start---
RLS=2;  % Velger Forgeting Factor
Pr=1000*eye(6);
lambda=0.92;
wb=2;
Threshold=5;

%Kjører system
disp('Kjører systemet...')
t=sim('ulin_RLS_sim');

%Skriver ut resultater for Forgeting Factor
disp('Skriver ut resultatet for Forgeting Factor')
Plot_idu(t,Y,Euler,Konst,Theta_hat,r2g)

%% Slutt
disp('Slutt')
```

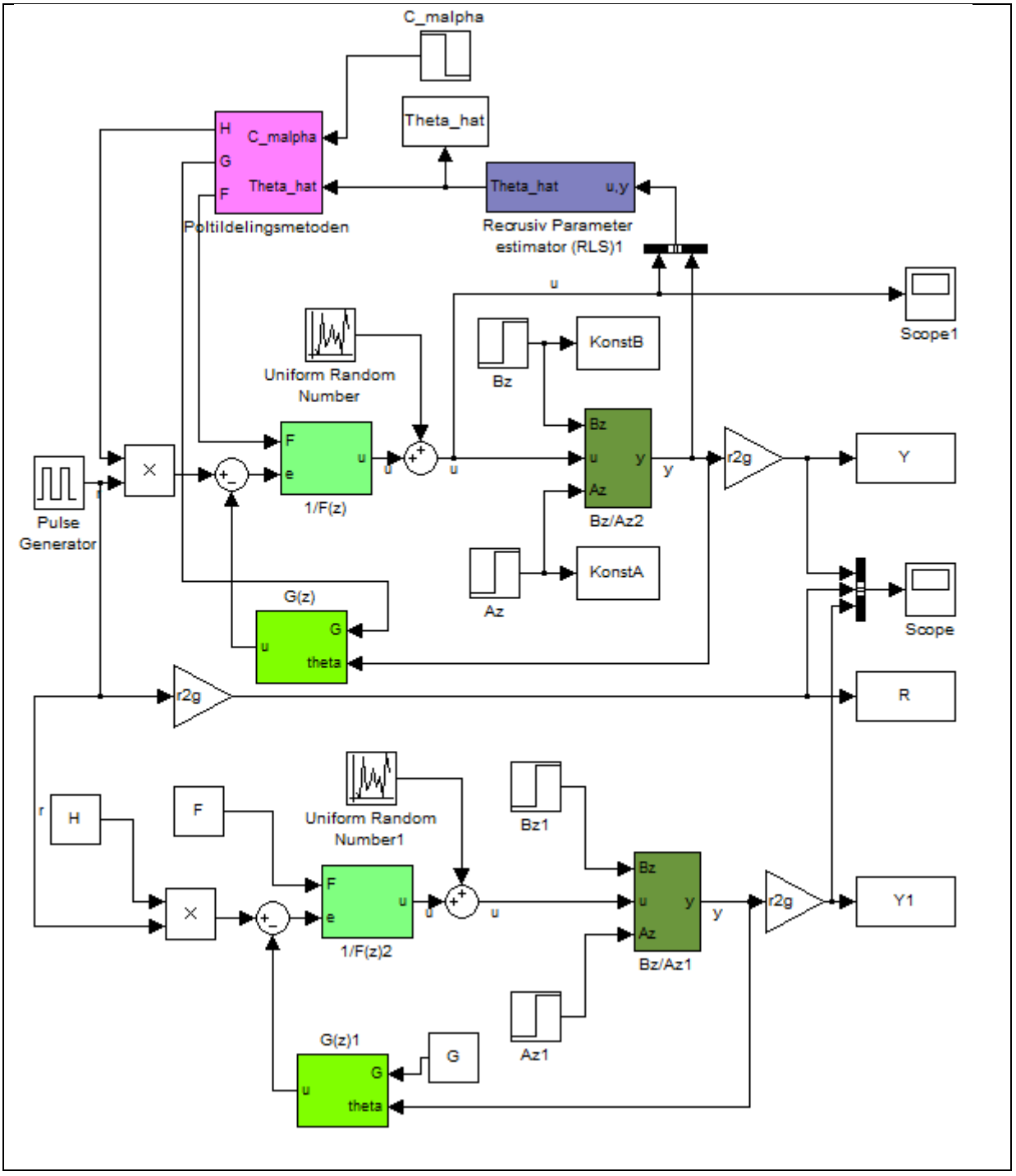
C.3.1.2 Plot for RLS skript fil

```
function Plot_idu(t,Y,Euler,Konst,Theta_hat,r2g)
%Dato: 20.05.10
%Av Thomas Algarheim
%   Plot for Ulin RLS Skript
figure
plot(t,Y(:,2).*r2g,'b',t,Y(:,3).*r2g,'r')
title('Angrepsvinkelen og sideslipsvinkelen')
legend('\alpha','\beta',4)
xlabel('Tid [s]')
ylabel('Vinkel [grader]')
grid on

figure
plot(t,Euler(1,:).*r2g,t,Euler(2,:).*r2g,t,Euler(3,:).*r2g)
title('Eulervinklene')
legend('Roll \Phi','Pitch \Theta', 'Yaw \Psi',4)
xlabel('Tid [s]')
ylabel('Vinkel [grader]')
grid on

figure
colors = hsv(3); labels = {'data'};
hold on
for d = 1:(3) %plot b
    subplot(3,1,d);plot(t,Konst(:,d)','r--'...
    ,t,Theta_hat(:,d),'b');
    if d==1
        title('A(z) koeffisientene')
    end
    labels{1} = ['a' int2str(d)];
    labels{2} = ['est.a' int2str(d)];
    legend(labels,-1);
end
figure
colors = hsv(3); labels = {'data'};
hold on
for d = 4:(6) %plot b
    subplot(3,1,(d-3));plot(t,Konst(:,d)','r--'...
    ,t,Theta_hat(:,d),'b');
    if d==4
        title('B(z) koeffisientene')
    end
    labels{1} = ['b' int2str(d-3)-1];
    labels{2} = ['est.b' int2str(d-3)-1];
    legend(labels,-1);
end
```

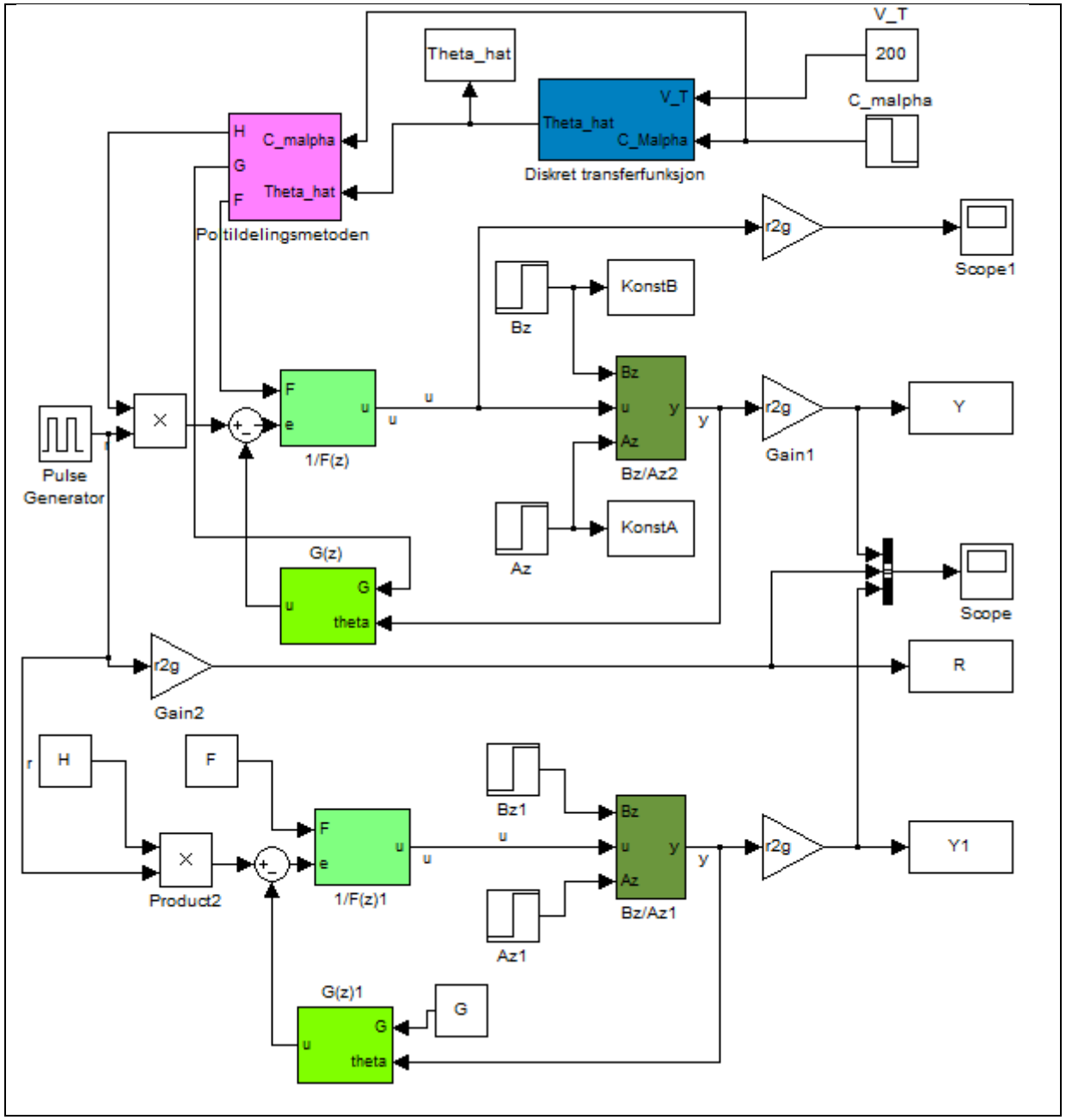
C.4 Kode kapitel 8 - Adaptiv regulering

C.4.1 Tester ut adaptive regulatorer på lineær modell

C.4.1.1 Simulinkmodell for RLS identifikasjon



C.4.1.2 Simulinkmodell for identifikasjon med diskretisering av forenklet system i pitch



C.4.1.3 Adaptiv reg. på lin. Mod. Skript

```
%-------Masteroppgave -------------
%Dato: 20.05.10
%Av Thomas Algarheim
%   Simulering av lineær fly modell med adaptiv regulator med  de
%   forskjellige identifiseringene: RLS, RLS med Random Walk, RLS med
%   Forgetting Factor og kontinuerlig diskretisering av den forenklede
pitch
%   systemet

clear all
clc
%% Initialisering
disp('Initialiserin')
```

```
Initialisering %i=2 variable aerodynamikk koeffisient C_M

Ts=0.01; %Sampel tiden
Time=60; %Lengde på simulasjon
step1=10;%Forandrer modell ved tidsskrit step1

disp('Finner diskret transferfunksjon for pitch')
V_T=200;
[Az(:,1),Bz(:,1)]=Dtf(V_T,C_malpha, Ts);
C_malpha=-0.1*1/g2r;
[Az(:,2),Bz(:,2)]=Dtf(V_T,C_malpha, Ts);

%Poltildeling
%Setter opp ønskelige poler
% syms sp
% Tp=sym2poly((sp-0.9802)*(sp-0.9802)*(sp-0.9802));%Ts=0.01;
Tp=[1.000   -2.940600000000018    2.882376120000000   -0.941768357608000];
Tp1=Tp;
[F G H]=PitchReg(Az,Bz,Tp);     %Setter Regulator etter ønskede poler

%% RLS
RLS=2;
Pr=100000*eye(6);
Rr=diag([500000 500000 0 0.09 60 0.01]);
lambda=0;
wb=3;
Threshold=5;
Theta_hat_init=[Az(2), Az(3), Az(4), Bz(1), Bz(2), Bz(3)]';

%Kjører systemet
disp('Kjører systemet...')
open_system('Adaptiv_reg_med_RLS_sim')
t=sim('Adaptiv_reg_med_RLS_sim');clear tout

% Sriver ut resultater
disp('plot')
Plot_Ladap(t,Y,Y1,R,KonstA,KonstB,Theta_hat,1)

%% RLS med Forgetting Factor ---start---
RLS=3;
Pr=1000*eye(6);
lambda=0.97;
wb=2;
Threshold=5;

%Kjører systemet
disp('Kjører systemet...')
t=sim('Adaptiv_reg_med_RLS_sim');clear tout

%Sriver ut resultater
disp('plot')
Plot_Ladap(t,Y,Y1,R,KonstA,KonstB,Theta_hat,2)

%% Setter poltildeling for id. med disk. av forenklet pitch system
[F G H]=PitchReg(Az,Bz,Tp);     %Setter Regulator etter ønskede poler
% Tp=sym2poly((sp-0.9602)*(sp-0.9602)*(sp-0.9502));%Ts=0.01;
% Tp1=sym2poly((sp-0.9802)*(sp-0.9802)*(sp-0.9802));%Ts=0.01;
Tp =[1.000   -2.870600000000025    2.746748120000000   -0.876069234808000];
Tp1=[1.000   -2.940600000000018    2.882376120000000   -0.941768357608000];
```

```
%Kjører systemet
disp('Kjører systemet...')
open_system('Adaptiv_reg_med_lin_tf_sim')
t=sim('Adaptiv_reg_med_lin_tf_sim');clear tout

%Sriver ut resultater
disp('plot')
Plot_Ladap(t,Y,Y1,R,KonstA,KonstB,Theta_hat,3)
```

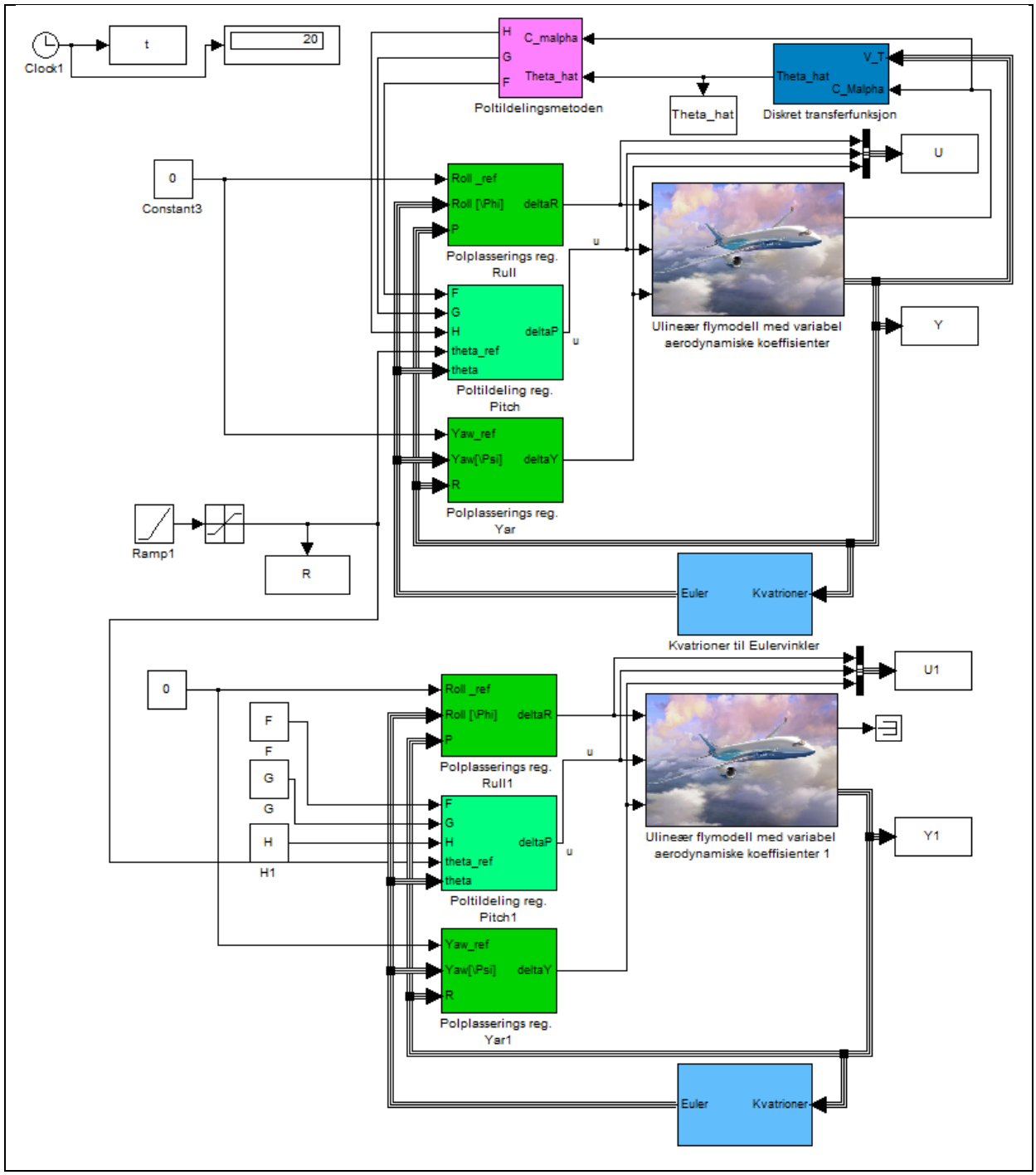
C.4.1.3 Adaptiv reg. på lin. Mod. Skript

```
function Plot_Ladap(t,Y,Y1,R,KonstA,KonstB,Theta_hat,i)
%Dato: 20.05.10
%Av Thomas Algarheim
%   Plot for Adaptiv_reg_paa_lin_mod_Skript

figure
plot(t,Y,'b',t,Y1,'r',t,R,'g'); axis([0 60 -2 16])
if i==1
    title('Apap. reg. med Random Walk  identifisering')
elseif i==2
    title('Apap. reg. med Forgetting Factor identifisering')
else
    title('Apap. reg. med d kontinuerlig identifisering med
diskretisering')
end
xlabel('Tid [s]')
ylabel('Vinkel [deg]')
legend('Ref. adaptiv regulator','Uten adaptiv regulator','Referanse')
grid on

figure;cla; %Figur A(z)
colors = hsv(3); labels = {'data'};
hold on
for d = 1:(3) %plot b
    subplot(3,1,d);plot(t,KonstA(:,d+1)'...
        ,t,Theta_hat(:,d),'color',colors(d,:));
    if d==1
        title('A(z) koeffisientene')
    end
    labels{1} = ['a' int2str(d)];
    labels{2} = ['est.a' int2str(d)];
    legend(labels,-1);
    grid on
 end
hold off

figure;cla; %Figur B(z)
colors = hsv(2+1); labels = {'data'};
hold on
for d = 1:(2+1) %plot b
    subplot(3,1,d);plot(t,KonstB(:,d)',...
        t,Theta_hat(:,d+3),'color',colors(d,:));
    if d==1
        title('B(z) koeffisientene')
    end
    labels{1} = ['b' int2str(d)-1];
    labels{2} = ['est.b' int2str(d)-1];
```

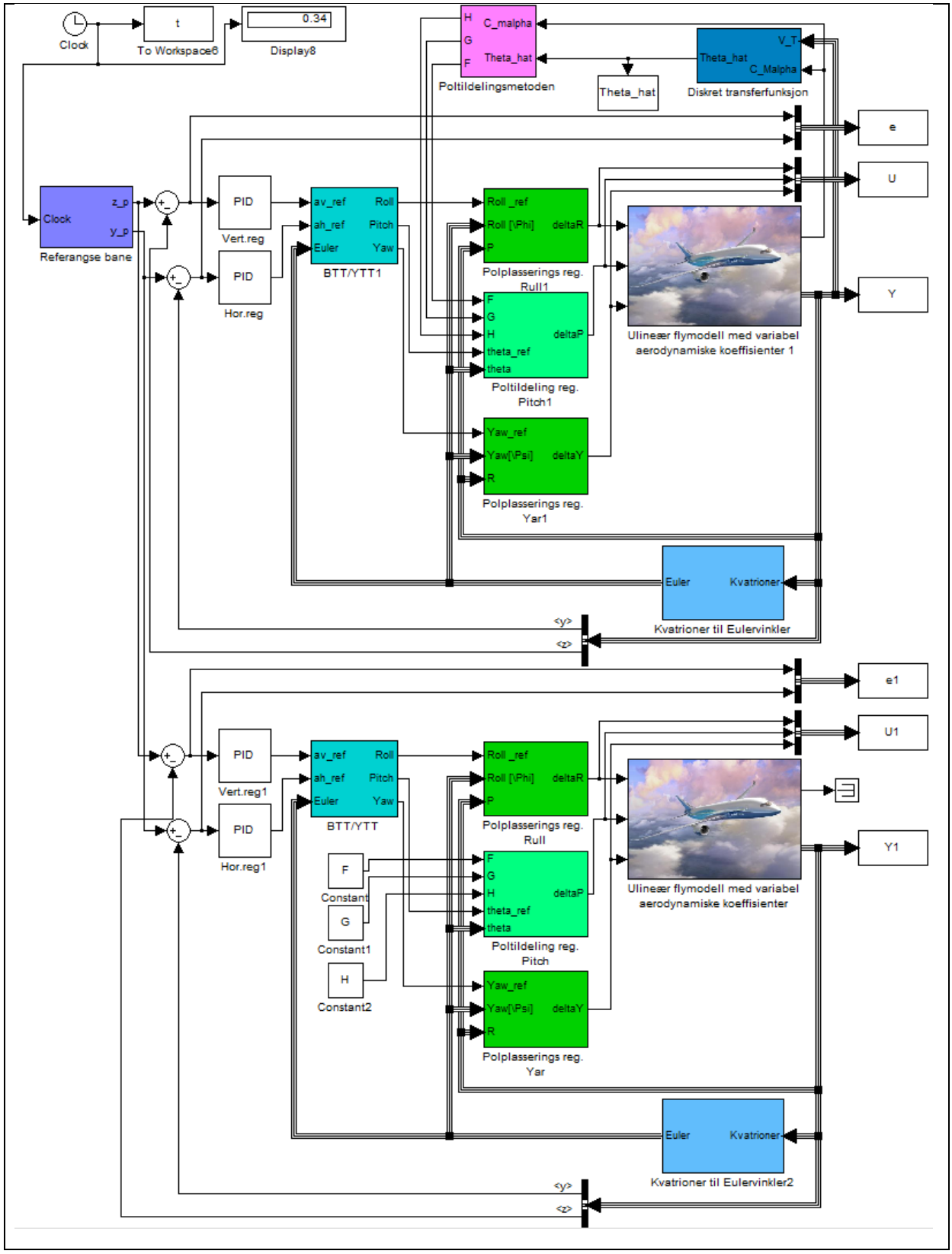
```
    legend(labels,-1);
    grid on
 end
hold off
```

C.4.2 Tester ut adaptive regulatorer på den ulineære modellen

C.4.2.1 Simulinkmodell av indre sløyfe



C.4.2.2 Simulinkmodell av fullt system



C.4.2.3 Adaptiv reg. på lin. Mod. Skript

```
%-------Masteroppgave 1.0.1 -------------
%Dato: 20.05.10
%Av Thomas Algarheim
```

```
%   Simulering av ulineær fly modell med kvatrioner, som vi regulerer med
%   adaptiv regulator på pitch sløyfen. Først simulerer vi de indre
%   sløyfene, deretter ytre sløyfe med step på referansen, slik at vi kan
%   innstille PID regulatorene med Zigler Niclors metode. Så simulerer vi
%   fyllt system som følger referansepunkter i rommet.

clc
clear all
%% Initialisering
disp('Initialiserin')
Initialisering %i=2 variable aerodynamikk koeffisient C_M

global Ts
Ts=0.01; %Sampel tiden
Time=20; %Lengde på simulasjon

%Startbetingelse
V_T=200;
 x0=[V_T,0*g2r,   0, 1, 0, 0, 0,0,0,0,0,0,0,200,0,0,]';
%x0=[V_T,alpha,beta,q0,q1,q2,q3,P,Q,R,x,y,z,  U,V,W,]'


%Lineariserer for indre sløyfe

disp('Lineariserer de tre indre sløyfene')
Linearisering_av_indre_loop

%Setter opp regulatorene
disp('Setter opp regulatorene')

%Regulator indre sløyfe
[Gr Gp Gy]=IndreReg(Ar, Br, Ap, Bp, Ay, By, 1,2);

%Reg Pitch
[Az,Bz]=Dtf(V_T,C_malpha, Ts);
%Finner lukket sløyfe poler
% syms sp
% Tp=sym2poly((sp-0.9602)*(sp-0.9602)*(sp-0.9502)); %Ts=0.01;
% Tp1=sym2poly((sp-0.9802)*(sp-0.9802)*(sp-0.9802));%Ts=0.01;
Tp =[1.000   -2.870600000000025    2.746748120000000   -0.876069234808000];
Tp1=[1.000   -2.940600000000018    2.882376120000000   -0.941768357608000];

[F G H]=PitchReg(Az,Bz,Tp);

%Kjører ulin. system med adaptiv indre reg.
disp('Kjører systemet...')
open_system('Adap_indre_ulin_sim')
t=sim('Adap_indre_ulin_sim');

%Skriver ut resultater
disp('Skriver ut resultatet fre indre sløyfe med adap. reg.')
Plot_ulin_ad(t,Y,U,Euler,Y1,U1,Euler1,Theta_hat,R,r2g)

%% Ytre sløyfe
% Regulator ytre sløyfe
Kpk_ver=0.021;               %Kritisk forsterkning vertikal reg.
Tp_ver=(8.23-7.31)/60;       %Kritisk periode vertikal reg.
[Kp_ver Ti_ver Td_ver Tf_ver]=YtreReg(Kpk_ver,Tp_ver,2);
Kp_ver=Kpk_ver*0.25;
```

```
Ti_ver=Tp_ver/1.8;

Kpk_hor=0.06;                %Kritisk forsterkning horisontal reg.
Tp_hor=(6.35-6.06)/60;       %Kritisk periode horisontal reg.
[Kp_hor Ti_hor Td_hor Tf_hor]=YtreReg(-Kpk_hor,-Tp_hor,2);

%Bank-To-Turn (BBT)/ Yaw-To-Trun (YTT)
% btt=1: BTT
% btt=0: YTT
btt=1;
Kbtt1=5;
Kbtt2=1;
Kbtt3=0.1;

% %Kjører ytre system
% disp('Kjører systemet...')
% open_system('tf_bas_ytre_loop_ZN_metode_sim')
% t=sim('tf_bas_ytre_loop_ZN_metode_sim');
%
% % Skriver ut resultater
% disp('Skriver ut resultatet fra yter sløyfe med adap. reg.')
% Plot_ytre(t,U,Y,Rh,Rv,r2g)

%% Referansepunkter
%Setter opp bane til flyet
disp('Finner bane ut i fra referanse punkter')
Time=35;
wpt_x = [0 100 200 600  1000  2000  3600      5000 5600    6000 6600]; %ca
200 m/s
wpt_y = [0 0    0   10    30     0  100       -10   -10  -10    -10];
wpt_z = [0 0    5   5   -10   -350  0     0     980 990   990];
wpt_t = [0 0.5  1   3    5     10    18      20       28  30 33];
[x_p y_p z_p tbane] = Bane(wpt_x, wpt_y, wpt_z, wpt_t, Ts, Time+1);

% Kjører fullt system med adaptiv regulator
disp('Kjører systemet...')
open_system('Adap_bane_ulin_sim')
t=sim('Adap_bane_ulin_sim');

% Skriver ut resultater
disp('Skriver ut resultatet med følging av ref. punkter')
Plot_fult_adap(t,Y,Y1,Euler,Euler1,U,U1,tbane,y_p,z_p,wpt_t...
    ,wpt_y,wpt_z,e,e1,Theta_hat,r2g)

%% Slutt
disp('Slutt')
```

C.4.2.4 Plot for indre sløyfe simulasjon

```
function Plot_ulin_ad(t,Y,U,Euler,Y1,U1,Euler1,Theta_hat,R,r2g)
%Dato: 20.05.10
%Av Thomas Algarheim
%   Plot for indre sløyfe med adaptiv regulator

figure
plot(t,Y(:,8)*r2g,'b',t,Y(:,9)*r2g,'r',t,Y(:,10)*r2g,'g')
title('Vinkelhastigherer')
legend('P','Q','R',4)
xlabel('Tid [s]')
```

```
ylabel('Vinkelhastighet[grader/s]')
grid on

figure
plot(t,Y(:,2).*r2g,t,Y(:,3).*r2g,t,Y1(:,2).*r2g,t,Y1(:,3).*r2g)
legend('\alpha adap','\beta adap','\alpha uten adap','\beta uten adap',4)
title('Angrepsvinkel og sideslipsvinkel')
xlabel('Tid [s]');ylabel('Vinkel [deg]')
grid on

figure
plot(t,U(:,2).*r2g,t,U1(:,2).*r2g)
legend('\deltaP Pitch med adap','\deltaP Pitch uten adap',1)
title('Pitch rorutslag ')
ylabel('Rorvinkel [deg]'), xlabel('Tid [s]');
grid on

figure
plot(t,Euler(2,:)*r2g,t,Euler1(2,:)*r2g,t,R*r2g)
legend('Respons med adaptiv reg.','Respons uten adaptiv reg.',...
    'Referangse',4)
title('Pitch respons')
xlabel('Tid [s]'); ylabel('Vinkel [deg]')
grid on

figure
 colors = hsv(3); labels = {'data'};
hold on
for d = 1:(3) %plot b
    subplot(3,1,d);plot(t,Theta_hat(:,d),'b');
    labels{1} = ['a' int2str(d)-1];
    legend(labels,-1);
end

figure(6)
colors = hsv(3); labels = {'data'};
hold on

for d = 4:(6) %plot b
    subplot(3,1,(d-3));plot(t,Theta_hat(:,d),'b');
    labels{1} = ['b' int2str(d-3)-1];
    legend(labels,-1);
end
```

C.4.2.5 Plot for fullt system med adaptiv regulator

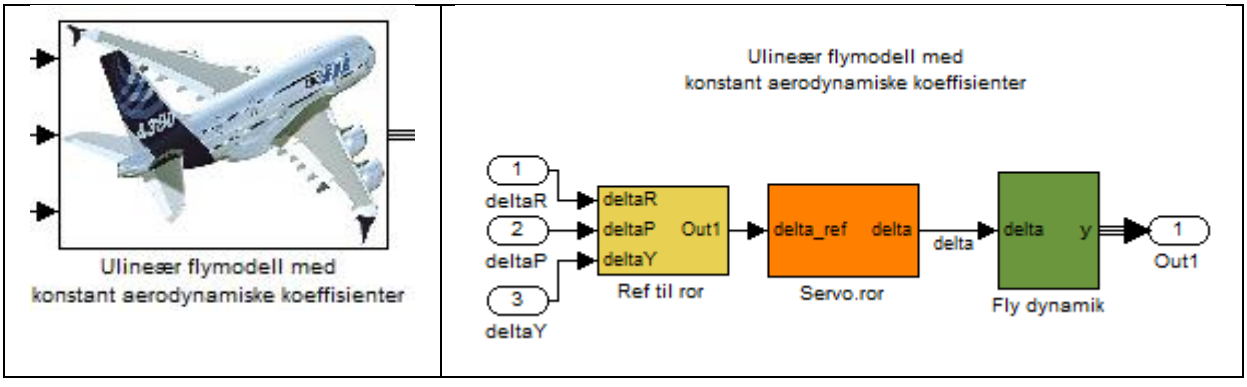
```
function Plot_fult_adap(t,Y,Y1,Euler,Euler1,U,U1,tbane,y_p,z_p,wpt_t...
    ,wpt_y,wpt_z,e,e1,Theta_hat,r2g)
%Dato: 20.05.10
%Av Thomas Algarheim
%   Plot for fullt system med adaptiv regulator

figure
plot(t,Y(:,2).*r2g,t,Y(:,3).*r2g,t,Y1(:,2).*r2g,t,Y1(:,3).*r2g)
axis([0 35 -40 20])
legend('\alpha,med adap','\beta,med adap','\alpha,uten adap',...
    '\beta,uten adap',4)
title('Angrepsvinkel og sideslipsvinkel')
xlabel('Tid [s]'); ylabel('Vinkel [deg]')
grid on
```

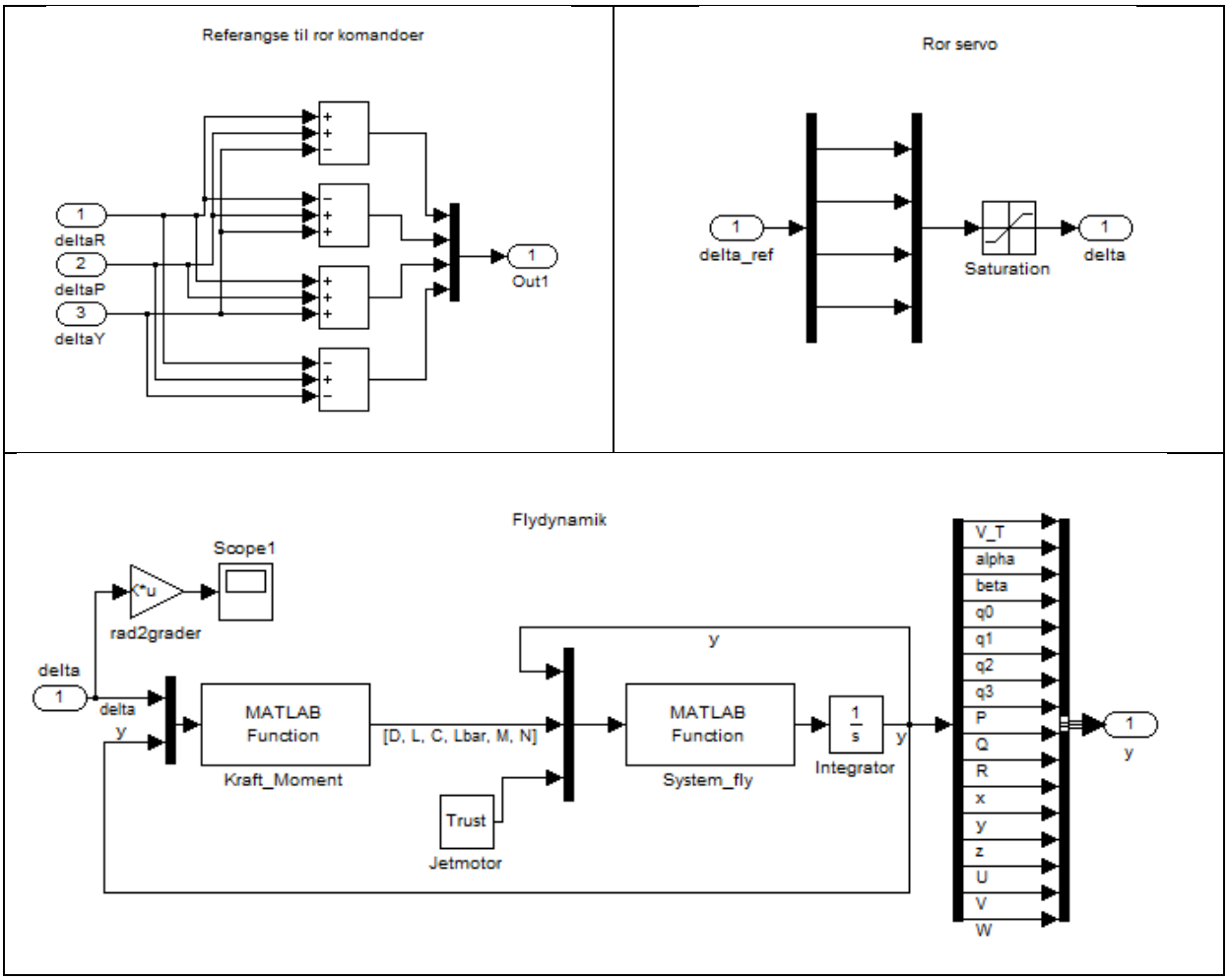
```
figure
plot(t,Euler(1,:).*r2g,t,Euler(2,:).*r2g,t,Euler(3,:).*r2g,...
    t,Euler1(1,:).*r2g,t,Euler1(2,:).*r2g,t,Euler1(3,:).*r2g)
axis([0 35 -45 70])
legend('Roll \Phi,med adap','Pitch \Theta,med adap',...
    'Yaw \Psi,med adap','Roll \Phi,uten adap','Pitch \Theta,uten adap',...
    'Yaw \Psi,uten adap',4)
title('Eulervinklene')
xlabel('Tid [s]'); ylabel('Vinkel [deg]')
grid on

figure
subplot(2,1,1);plot(t,U(:,1).*r2g,t,U(:,2).*r2g,t,U(:,3).*r2g)
axis([0 35 -25 80])
legend('\deltaR Roll','\deltaP Pitch','\deltaY Ya',1)
title('Rorutslag med adaptiv regulator')
xlabel('Tid [s]'); ylabel('Rorvinkel [deg]')
grid on
subplot(2,1,2);plot(t,U1(:,1).*r2g,t,U1(:,2).*r2g,t,U1(:,3).*r2g)
axis([0 35 -25 80])
legend('\deltaR Roll','\deltaP Pitch','\deltaY Yaw',1)
title('Rorutslag uten adaptiv regulator')
xlabel('Tid [s]'); ylabel('Rorvinkel [deg]')
grid on

figure
subplot(2,1,1);plot(t,Y(:,12),'b',t,Y1(:,12),'r',tbane,y_p,'g',...
    wpt_t,wpt_y,'o');grid on
axis([0 35 -50 150])
title('Posisjon horisontalt')
ylabel('y posisjon [m]'), xlabel('Tid [s]');            %axis([0 15000 -2
10 -5 20])
legend('Banen til fly','Uten adap','Ref. bane','Ref. punkter',1)
subplot(2,1,2);plot(t,Y(:,13),'b',t,Y1(:,13),'r',tbane,z_p,'g',...
    wpt_t,wpt_z,'o')
axis([0 35 -500 1500])
title('Posisjon vertikalt')
ylabel('z posisjon [m]'), xlabel('Tid [s]');
legend('Banen til fly','Uten adap','Ref. bane','Ref. punkter',4)
grid on

figure
plot3(t,Y(:,12),Y(:,13),'b',t,Y1(:,12),Y1(:,13),'g',wpt_t,wpt_y,wpt_z,...
    'o',tbane,y_p,z_p,'r-.')
axis([0 35 -50 150 -500 1500])
legend('Banen til fly med adap','Banen til fly uten adap',...
    'Ref. punkter','Ref. bane',1)
title('3D plot av banen til flyet')
xlabel('Tid [s]')
ylabel('y posisjon [m].')
zlabel('z posisjon [m]')
grid on

figure
subplot(2,1,1);plot(t,e(:,1),t,e1(:,1))
axis([0 35 -100 140])
legend('Avvik fra banen med adaptiv reg',...
    'Avvik fra banen uten adaptiv reg',1)
title('Feil i vertikal retning')
```

```
xlabel('Tid [s]'); ylabel('Feil [m]')
grid on
subplot(2,1,2);plot(t,e(:,2),t,e1(:,2))
axis([0 35 -15 20])
legend('Avvik fra banen med adaptiv reg',...
    'Avvik fra banen uten adaptiv reg',1)
title('Feil i horisontalt retning')
xlabel('Tid [s]'); ylabel('Feil [m]')
grid on

figure
colors = hsv(3); labels = {'data'};
hold on
for d = 1:(3) %plot b
    subplot(3,1,d);plot(t,Theta_hat(:,d),'b');
    if d==1
        title('A(z) koeffisientene')
    end
    labels{1} = ['a' int2str(d)-1];
    legend(labels,-1);
    grid on
end
hold off

figure
colors = hsv(3); labels = {'data'};
hold on
for d = 4:(6) %plot b
    subplot(3,1,(d-3));plot(t,Theta_hat(:,d),'b');
    if d==4
        title('B(z) koeffisientene')
    end
    labels{1} = ['b' int2str(d-3)-1];
    legend(labels,-1);
    grid on
end
hold off
```

C.5 Felles blokker og embedded Matlab funksjoner for Simulink

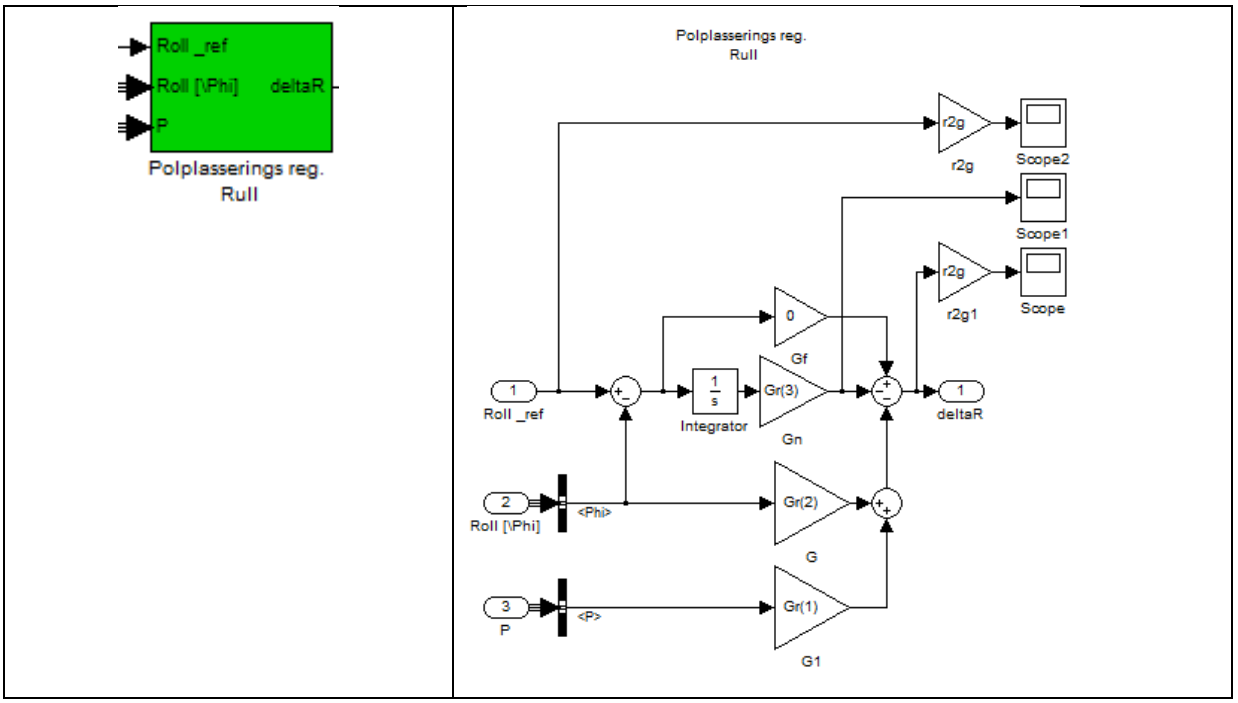
C.5.1 Ulineær flymodell med konstant aerodynamiske koeffisienter



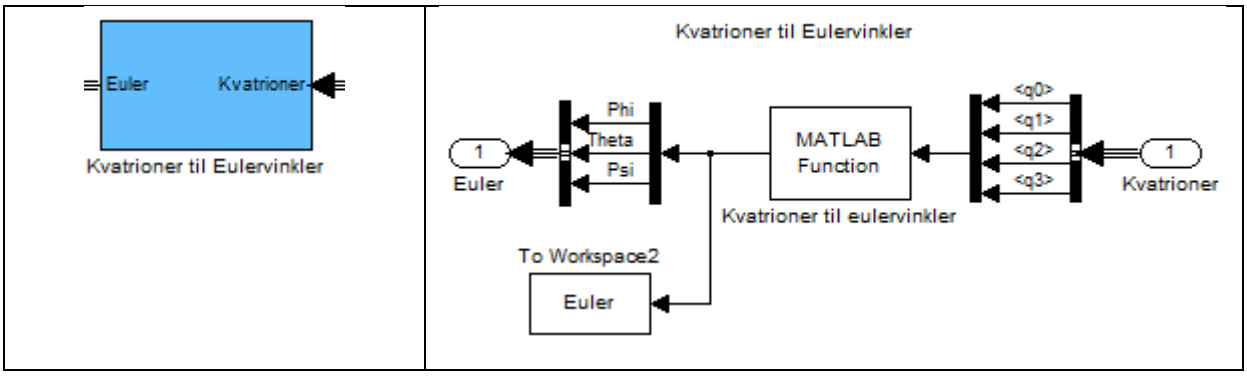


C.5.2 Regulator med polplasseringsmetoden

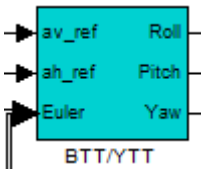
Her er det tatt med roll regulator med polplasseringsmetoden, de andre to metodene er eksakt de samme bortsett fra G parameteren.

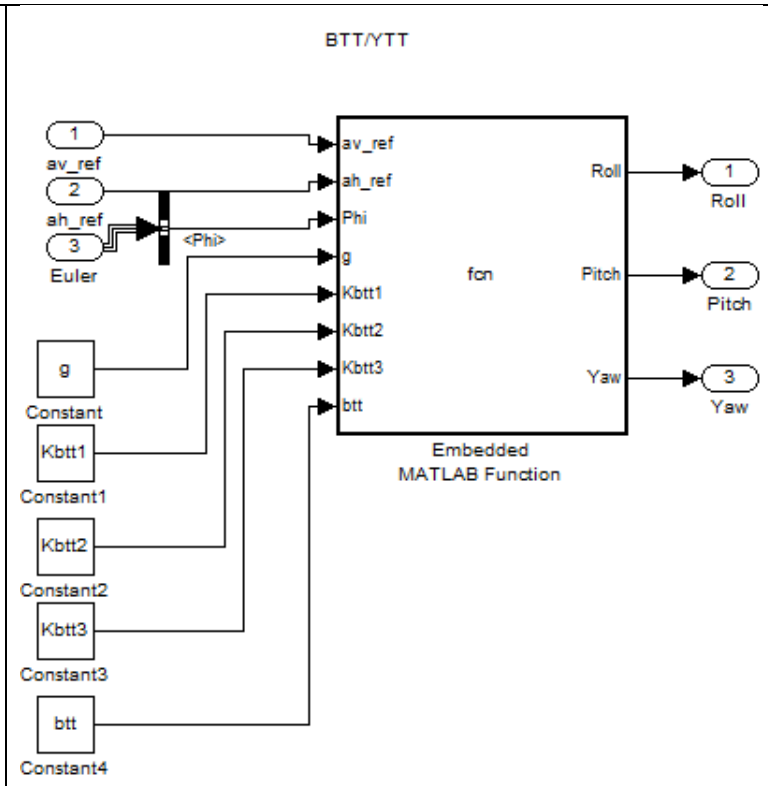


C.5.3 Kvatrioner til Eulervinkler



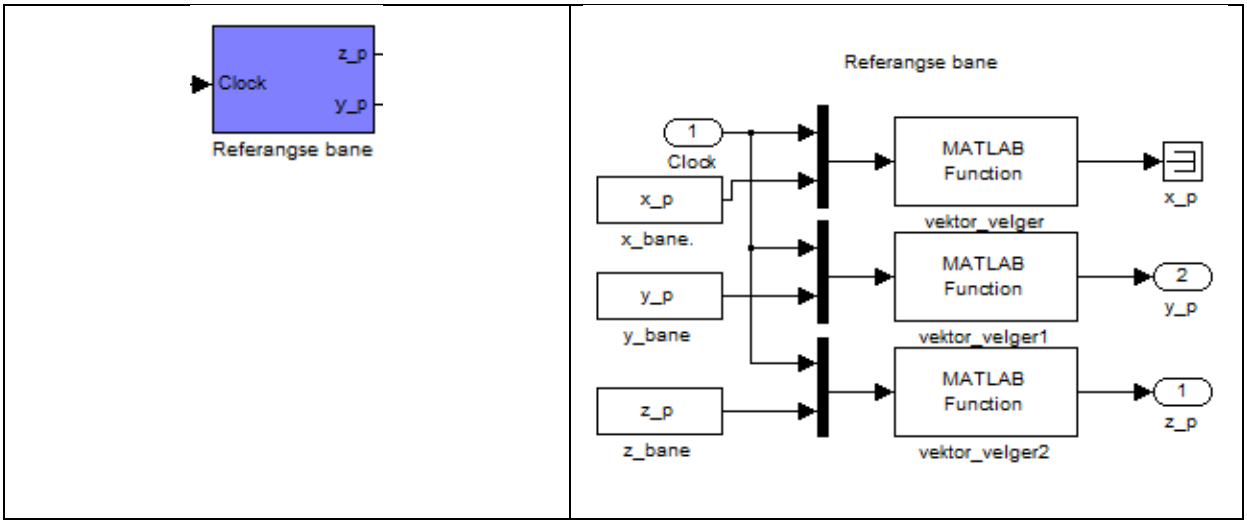
C.5.3 Bank-To-Turn / Yaw-To-Turn



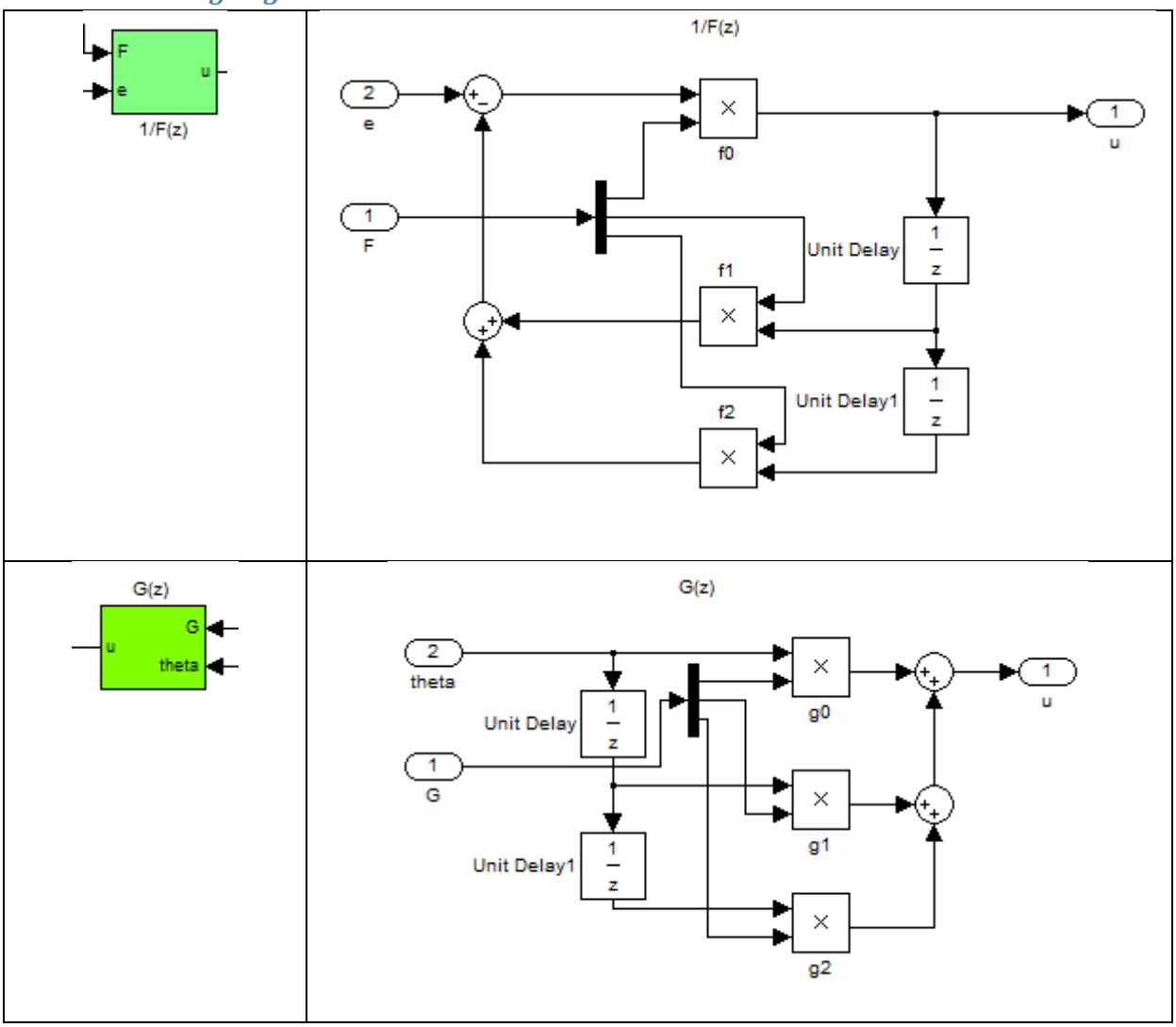



```
function [Roll,Pitch,Yaw] = fcn(av_ref,ah_ref,Phi,g,Kbtt1,Kbtt2,Kbtt3,btt)
if btt==1
    %BTT
    av_refM=g-av_ref;
    Roll = Kbtt1*atan2(ah_ref,av_refM);
    Pitch= Kbtt2*(av_ref*cos(Phi)-ah_ref*sin(Phi));
    Yaw  = Kbtt3*(ah_ref*cos(Phi)+av_refM*sin(Phi));
else
    %YTT
    Roll=0; %Roll
    Pitch=av_ref;%Pitch
    Yaw=ah_ref;%Yaw
end
```

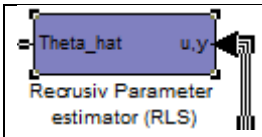
C.5.5 Referangsebane

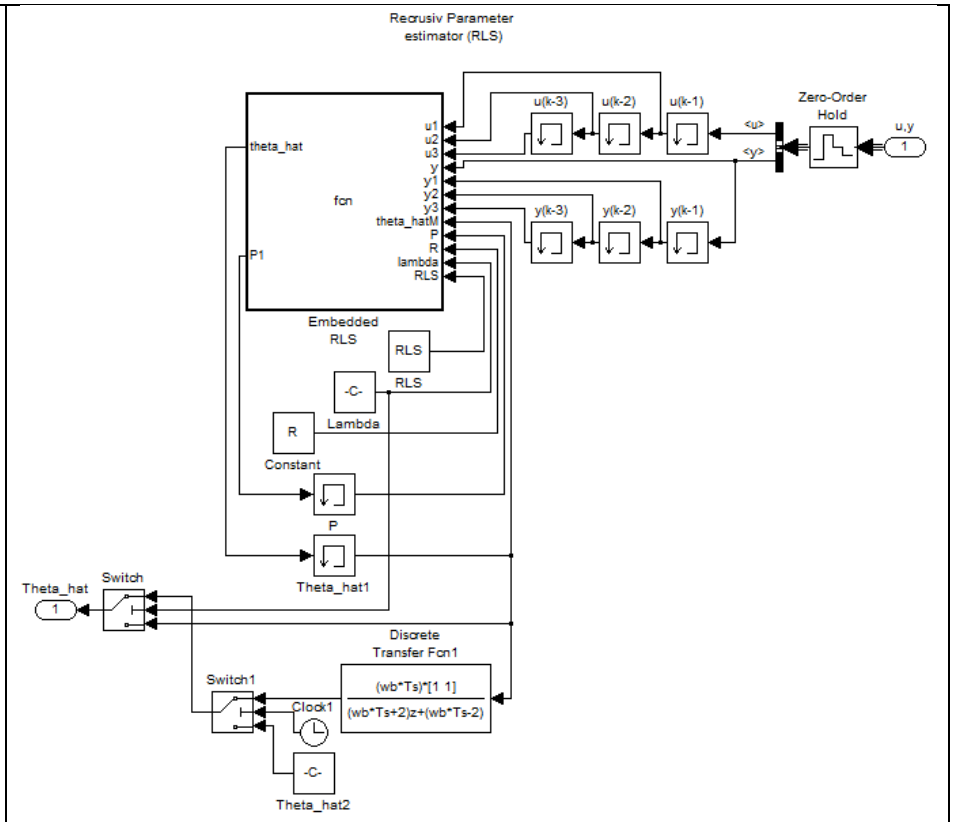


C.5.6 Poltildelingsregulator



C.5.6 Recrusiv Parameter estimator (RLS)

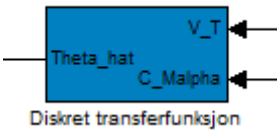


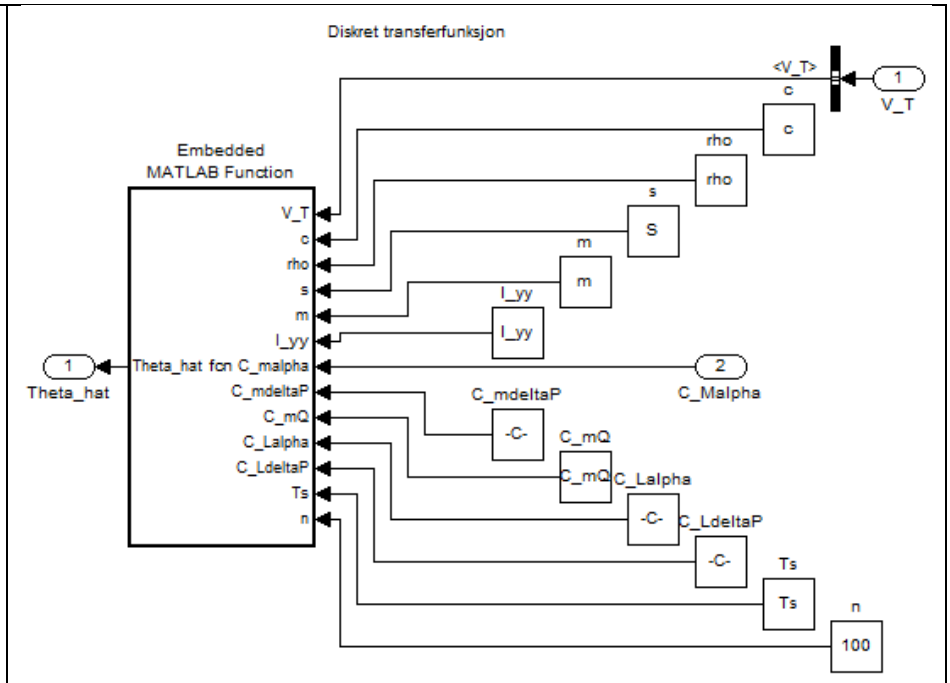



```
function [theta_hat, P1] =
fcn(u1,u2,u3,y,y1,y2,y3,theta_hatM,P,R,lambda,RLS)

%Dato: 19.05.10

%Av Thomas Algarheim

%   RLS Algoritmen

if RLS==1

    %RLS Algoritmen

    Im=diag(ones(1,6));

    fi=[-y1, -y2, -y3, u1, u2, u3]';

    e=y-fi'*theta_hatM;

    K=(P*fi)/(1+fi'*P*fi);

    theta_hat=theta_hatM+K*e;

    P1=P*(Im-((fi*fi'*P)/(1+fi'*P*fi)));

elseif RLS==2

    %Random Walk

    Im=diag(ones(1,6));

    fi=[-y1, -y2, -y3, u1, u2, u3]';
```

```
    e=y-fi'*theta_hatM;
    K=(P*fi)/(1+fi'*P*fi);
    theta_hat=theta_hatM+K*e;
    P1=P*(Im-((fi*fi'*P)/(1+fi'*P*fi)))+R;
elseif RLS==3
    %Forgeting factor
    Im=diag(ones(1,6));
    fi=[-y1, -y2, -y3, u1, u2, u3]';
    e=y-fi'*theta_hatM;
    K=(P*fi)/(lambda+fi'*P*fi);
    theta_hat=theta_hatM+K*e;
    P1=(1/lambda)*P*(Im-((fi*fi'*P)/(lambda+fi'*P*fi)));
else
    %Konstant trace
    Im=diag(ones(1,6));
    fi=[-y1, -y2, -y3, u1, u2, u3]';
    e=y-fi'*theta_hatM;
    K=(P*fi)/(1+fi'*P*fi);
    theta_hat=theta_hatM+K*e;
    Rt=(fi'*P^2*fi)/(1+fi'*P*fi);
    R=1/6*Rt*Im;
    P1=P*(Im-((fi*fi'*P)/(1+fi'*P*fi)))+R;
end
```

C.5.7 Diskret transferfunksjon



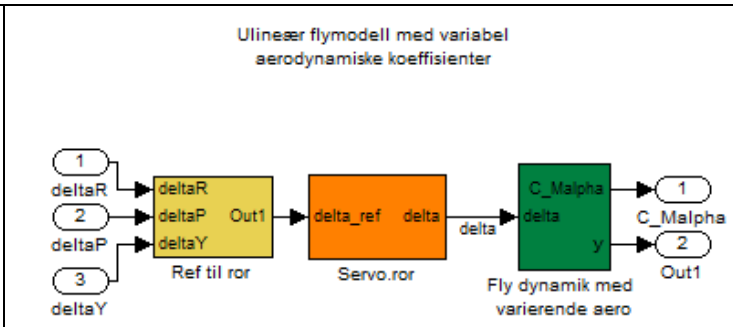


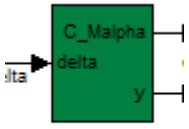

```
function Theta_hat =
fcn(V_T,c,rho,s,m,I_yy,C_malpha,C_mdeltaP,C_mQ,C_Lalpha,C_LdeltaP,Ts,n)

%#eml


K1=0.5*rho*s*V_T^2*c*C_mdeltaP;

K2=0.5*rho*s*V_T^2*c*C_malpha;

K3=1/(2*m)*rho*s*V_T*(C_Lalpha);

K4=0.25*rho*s*V_T*c^2*C_mQ;

K5=1/(2*m)*rho*s*V_T*(C_LdeltaP);


Ap=[-(K3)        K3           0;

    0            0            1;

    -(K2)/I_yy   (K2)/I_yy    K4/I_yy];

Bp=[K5           0            K1/I_yy]';

Cp=[0            1            0];


Phi=zeros(size(Ap));

Delta=zeros(size(Bp));

%Hjelpevariabler
```

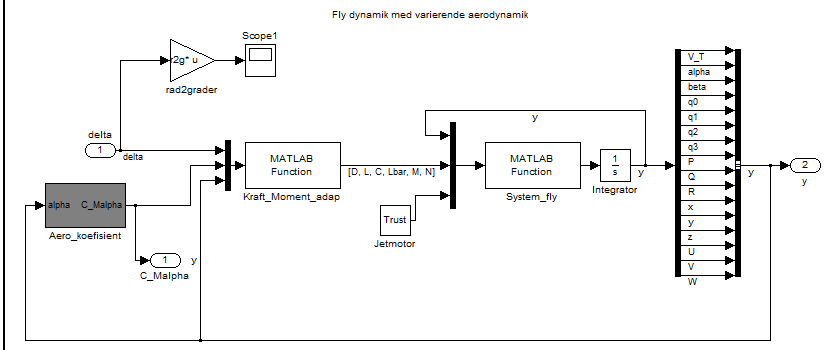
```
A_Phi=Ap*Ts;
Fi_2=zeros(size(Ap));
F_t1=eye(size(Ap));
F_t2=eye(size(Ap));
k=1;
for j=1:n
    Phi=Phi+F_t1;
    Fi_2=Fi_2+F_t2;
    F_t1=A_Phi*F_t1/k;
    F_t2=A_Phi*F_t2/(k+1);
    k=k+1;
end
Delta=Fi_2*Bp*Ts;

b0=-(-Cp(2)*Delta(2));

b1=-(Cp(2)*(Delta(2)*Phi(1,1)+Delta(2)*Phi(3,3)-Delta(1)*Phi(2,1)-...
    Delta(3)*Phi(2,3)));

b2=-(Cp(2)*(...
   Delta(1)*(Phi(2,1)*Phi(3,3)-Phi(3,1)*Phi(2,3))+...
   Delta(2)*(Phi(1,3)*Phi(3,1)-Phi(1,1)*Phi(3,3))+...
   Delta(3)*(Phi(1,1)*Phi(2,3)-Phi(2,1)*Phi(1,3))));

a1=-(Phi(1,1)+Phi(2,2)+Phi(3,3));

a2=-(Phi(1,2)*Phi(2,1)+Phi(1,3)*Phi(3,1)+Phi(2,3)*Phi(3,2)...
   -Phi(1,1)*(Phi(2,2)+Phi(3,3))-Phi(2,2)*Phi(3,3));

a3=-(Phi(1,1)*(Phi(2,2)*Phi(3,3)-Phi(2,3)*Phi(3,2))...
  +Phi(1,2)*(Phi(3,1)*Phi(2,3)-Phi(2,1)*Phi(3,3))...
  +Phi(1,3)*(Phi(2,1)*Phi(3,2)-Phi(2,2)*Phi(3,1)));
```

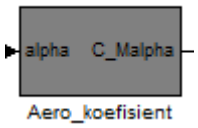
```
Theta_hat = [a1 a2 a3 b0 b1 b2]';
```

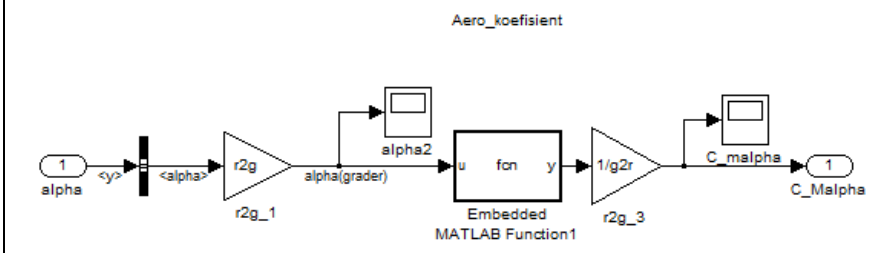
C.5.8 Ulineær flymodell med variable aerodynamiske koeffisienter





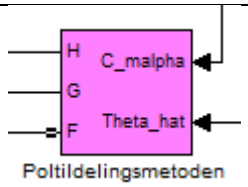


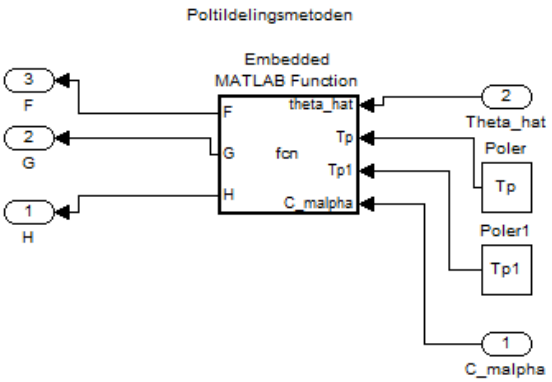





```
function y = fcn(u)


if -8 > u

    y=-0.1 ;

elseif 8<u

    y=-0.1;

else

    y=0.05;

end
```

C.5.9 Poltildelingsmetoden

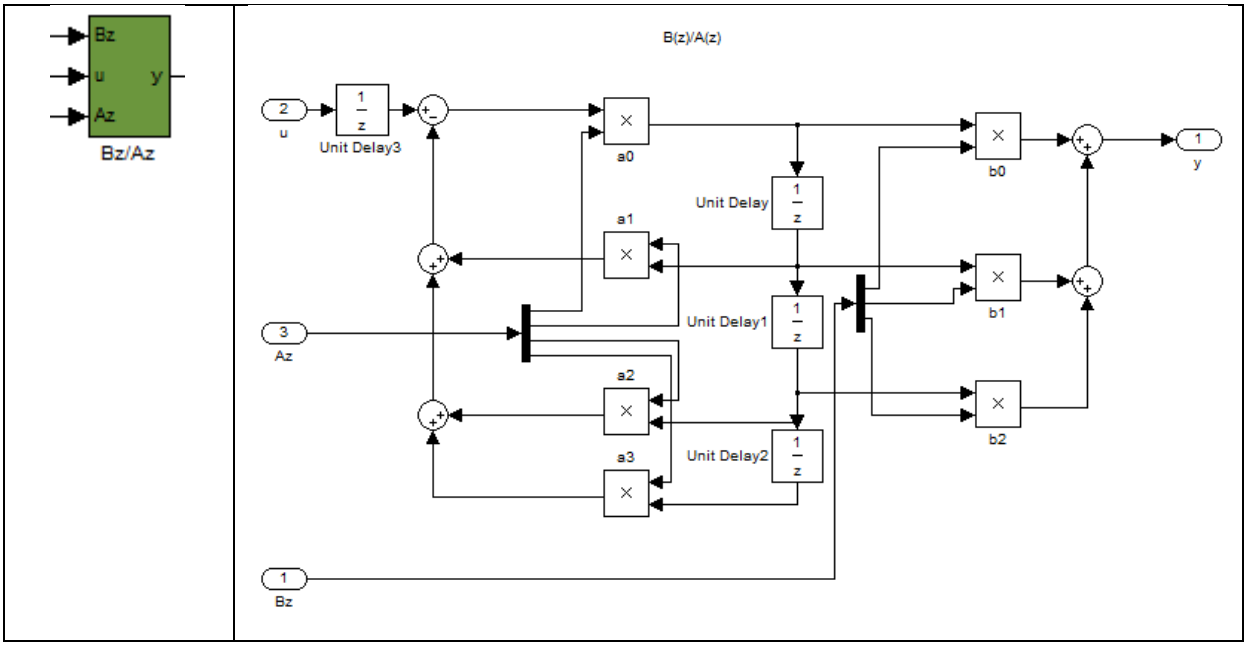




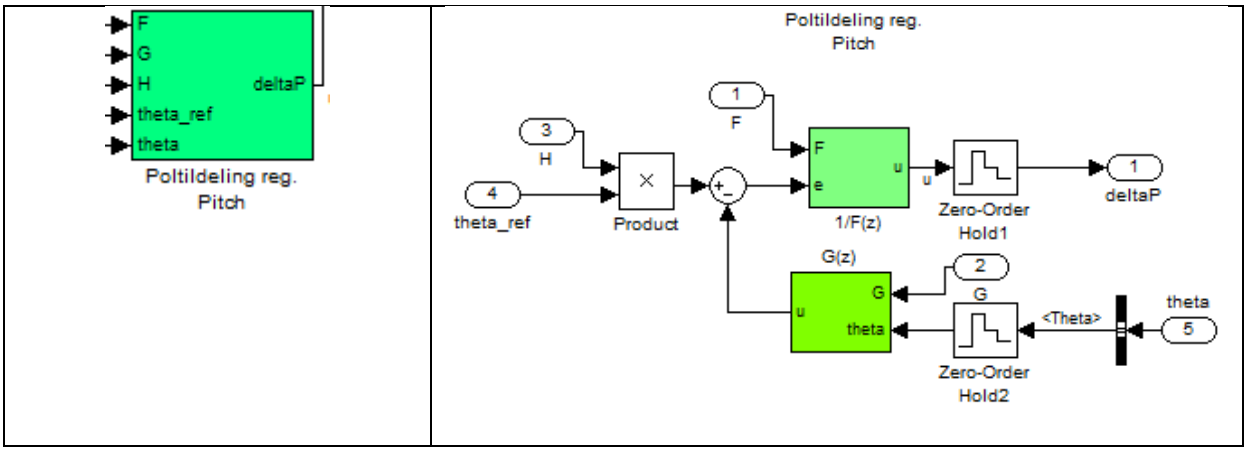

```
function [F,G,H] = fcn(theta_hat,Tp,Tp1,C_malpha)
%#eml
nf=2;


if C_malpha < 0.000
        Tp=Tp1;
end



a1=theta_hat(1);a2=theta_hat(2);a3=theta_hat(3);
b0=theta_hat(4);b1=theta_hat(5);b2=theta_hat(6);


t1=Tp(2);t2=Tp(3);t3=Tp(4);


H=(1+t1+t2+t3)/(b0+b1+b2);%1/(b0+b1+b2)%


Ar=[1   0    b0 0    0   ;     %[f1]
    a1 1    b1 b0   0   ;     %[f2]
    a2 a1   b2 b1   b0 ;     %[g0]
    a3 a2   0  b2   b1 ;     %[g1]
    0  a3   0  0    b2];     %[g2]
Br=[(t1-a1) (t2-a2) (t3-a3) 0 0]';
```

```
theta_r=Ar\Br; %[f1 f2 g0 g1 g2]'


F=[1 theta_r(1:nf)']';

G=theta_r(nf+1:length(theta_r));
```

C.5.9 Bz/Az



C.5.10 Poltildeling reg. Pitch



C.6 M.filer og funksjoner som er felles

C.6.1 Initalisering

```
% Initialisering
%Dato: 18.5.10
%Av Thomas Algarheim
%    Initialisering for flymodell
%    i=1 er for konstante aerodynamiske koeffisienter
%    i=2 er for variable aerodynamisk koeffisient C_M
```

```
clear Trust I_xx I_yy I_zz I_xz m
clear rho S c b
clear C_D0 C_Dalpha2
clear C_Lalpha C_LQ C_Ldelta1 C_Ldelta2 C_Ldelta3 C_Ldelta4
clear C_LdeltaR C_LdeltaP C_LdeltaY
clear C_Cbeta C_CR  C_Cdelta1 C_Cdelta2 C_Cdelta3 C_Cdelta4
clear C_CdeltaR C_CdeltaP C_CdeltaY
clear C_lbeta C_lP C_lR C_ldelta1 C_ldelta2 C_ldelta3 C_ldelta4
clear C_ldeltaR C_ldeltaP C_ldeltaY
clear C_m0 C_malpha C_mQ C_mdelta1 C_mdelta2 C_mdelta3 C_mdelta4
clear C_mdeltaR C_mdeltaP C_mdeltaY
clear C_nbeta C_nP C_nR C_ndelta1 C_ndelta2 C_ndelta3 C_ndelta4
clear C_ndeltaR C_ndeltaP C_ndeltaY
global Trust I_xx I_yy I_zz I_xz m g
global rho S c b
global C_D0 C_Dalpha2
global C_Lalpha C_LQ C_Ldelta1 C_Ldelta2 C_Ldelta3 C_Ldelta4
global C_LdeltaR C_LdeltaP C_LdeltaY
global C_Cbeta C_CR C_Cdelta1 C_Cdelta2 C_Cdelta3 C_Cdelta4
global C_CdeltaR C_CdeltaP C_CdeltaY
global C_lbeta C_lP C_lR C_ldelta1 C_ldelta2 C_ldelta3 C_ldelta4
global C_ldeltaR C_ldeltaP C_ldeltaY
global C_m0 C_malpha C_mQ C_mdelta1 C_mdelta2 C_mdelta3 C_mdelta4
global C_mdeltaR C_mdeltaP C_mdeltaY
global C_nbeta C_nP C_nR C_ndelta1 C_ndelta2 C_ndelta3 C_ndelta4
global C_ndeltaR C_ndeltaP C_ndeltaY

i=2;
%Flyet
Trust=3300;          %[Nm]
I_xx=7.3243;
I_yy=177.3608;
I_zz=177.9664;
I_xz=1;
m=325.7;             %[kg]
g=9.81;              %[m/s^2]

%Aerodynamiske konstanter
rho=1.2928;          %[] trykk
S=0.75;              %[m^2]
c=0.47;              %[m^2]
b=1.3;               %[m^2]

r2g=180/pi; %grader til radianer
g2r=pi/180; %radianer til grader

%Drag
C_D0=0.1719;         %[]
C_Dalpha2=0.0057;    %[1/rad]

%Lift
C_Lalpha=3.8969;     %[1/rad]
% C_Lalphadot=0;       %[1/rad]
C_LQ=0;              %[1/rad]
C_Ldelta1=0.1998;    %[1/rad]
C_Ldelta2=0.1998;    %[1/rad]
C_Ldelta3=0.1998;    %[1/rad]
C_Ldelta4=0.1998;    %[1/rad]
C_LdeltaR=C_Ldelta1-C_Ldelta2+C_Ldelta3-C_Ldelta4;
C_LdeltaP=C_Ldelta1+C_Ldelta2+C_Ldelta3+C_Ldelta4;
```

```
C_LdeltaY=-C_Ldelta1+C_Ldelta2+C_Ldelta3-C_Ldelta4;

%Sidekraft
C_Cbeta=0.8305;      %[1/rad]
% C_Cbetadot=0;        %[1/rad]
C_CR=0;              %[1/rad]
C_Cdelta1=-0.1677;   %[1/rad]
C_Cdelta2=0.1677;    %[1/rad]
C_Cdelta3=0.1677;    %[1/rad]
C_Cdelta4=-0.1677;   %[1/rad]
C_CdeltaR=C_Cdelta1-C_Cdelta2+C_Cdelta3-C_Cdelta4;
C_CdeltaP=C_Cdelta1+C_Cdelta2+C_Cdelta3+C_Cdelta4;
C_CdeltaY=-C_Cdelta1+C_Cdelta2+C_Cdelta3-C_Cdelta4;

%Rull moment
C_lbeta=-0.1411;     %[1/rad]
C_lP=-1.4567;        %[1/rad]
C_lR=-1.1827;        %[1/rad]
C_ldelta1=0.1394;    %[1/rad]
C_ldelta2=-0.1394;   %[1/rad]
C_ldelta3=0.1603;    %[1/rad]
C_ldelta4=-0.1603;   %[1/rad]
C_ldeltaR=C_ldelta1-C_ldelta2+C_ldelta3-C_ldelta4;
C_ldeltaP=C_ldelta1+C_ldelta2+C_ldelta3+C_ldelta4;
C_ldeltaY=-C_ldelta1+C_ldelta2+C_ldelta3-C_ldelta4;

%Pitc moment
if i == 1
    C_m0=0;              %[.] ingen benevning
    C_malpha=-0.3951;    %[1/rad]
    % C_malphadot=0;       %[1/rad]
    C_mQ=-15.4152;       %[1/rad]
    C_mdelta1=-0.4507;   %[1/rad]
    C_mdelta2=-0.4507;   %[1/rad]
    C_mdelta3=-0.4507;   %[1/rad]
    C_mdelta4=-0.4507;   %[1/rad]
else
    C_m0=0;               %[.] ingen benevning
    C_malpha=0.05*1/g2r;     %[1/rad] for alpha -8-8 deg
%     C_malphadot=0;          %[1/rad]
    C_mQ=-15.4152;           %[1/rad]
    C_mdelta1=-0.02*1/g2r;   %[1/rad]
    C_mdelta2=-0.02*1/g2r;   %[1/rad]
    C_mdelta3=-0.02*1/g2r;   %[1/rad]
    C_mdelta4=-0.02*1/g2r;   %[1/rad]
end
C_mdeltaR=C_mdelta1-C_mdelta2+C_mdelta3-C_mdelta4;
C_mdeltaP=C_mdelta1+C_mdelta2+C_mdelta3+C_mdelta4;
C_mdeltaY=-C_mdelta1+C_mdelta2+C_mdelta3-C_mdelta4;

%Yawing moment
C_nbeta=0.0457;      %[1/rad]
C_nP=-5.6989;        %[1/rad]
C_nR=-7.9635;        %[1/rad]
C_ndelta1=-0.3781;   %[1/rad]
C_ndelta2=0.3781;    %[1/rad]
C_ndelta3=0.3781;    %[1/rad]
C_ndelta4=-0.3781;   %[1/rad]
C_ndeltaR=C_ndelta1-C_ndelta2+C_ndelta3-C_ndelta4;
C_ndeltaP=C_ndelta1+C_ndelta2+C_ndelta3+C_ndelta4;
C_ndeltaY=-C_ndelta1+C_ndelta2+C_ndelta3-C_ndelta4;
```

C.6.2 Linearisering av indre sløyfe

```
%Linearisering_av_indre_loop

%Dato: 23.04.10
%Av Thomas Algarheim
%   Lineariserer med en vindhastighet på  V_T=200[m/s] og en angrepsvinkel
%   på alpha=5[deg] og vi  får systemene
%   System i roll, Ar og Br
%   System i pitch, Ap og Bp
%   System i yaw, Ay og By

global I_xx I_yy I_zz I_xz
global rho S c b
global C_D0 C_Dalpha2
global C_Lalpha
global C_LdeltaR C_LdeltaP C_LdeltaY
global C_Cbeta
global C_CdeltaR C_CdeltaP C_CdeltaY
global C_lbeta C_lP C_lR
global C_ldeltaR C_ldeltaP C_ldeltaY
global C_m0 C_malpha C_mQ
global C_mdeltaR C_mdeltaP C_mdeltaY
global C_nbeta C_nP C_nR
global C_ndeltaR C_ndeltaP C_ndeltaY

syms V_T alpha beta P Q R  phi theta psi real
syms deltaR deltaP deltaY  real

%% Krefter og moment
%Finner kreftene og momentene på flyet ved arbeidspunkt

qbar=0.5*rho*V_T^2; %Dynamisk trykk

%Drag
D=qbar*S*(C_D0+C_Dalpha2*alpha^2);
%Lift
L=qbar*S*(C_Lalpha*alpha+c/V_T*(C_LQ*Q)+...
    C_LdeltaR*deltaR+C_LdeltaP*deltaP+C_LdeltaY*deltaY);
%Sidekraft
C=qbar*S*(C_Cbeta*beta+b/V_T*(C_CR*R)+...
    C_CdeltaR*deltaR+C_CdeltaP*deltaP+C_CdeltaY*deltaY);
%Rull moment
Lbar=qbar*S*b*(C_lbeta*beta+b/V_T*(C_lP*P+C_lR*R)+...
   C_ldeltaR*deltaR+C_ldeltaP*deltaP+C_ldeltaY*deltaY);
%Pitc moment
M=qbar*S*c*(C_m0+C_malpha*alpha+c/V_T*(C_mQ*Q)+...
    C_mdeltaR*deltaR+C_mdeltaP*deltaP+C_mdeltaY*deltaY);
%yawing moment
N=qbar*S*b*(C_nbeta*beta+b/V_T*(C_nP*P+C_nR*R)+...
    C_ndeltaR*deltaR+C_ndeltaP*deltaP+C_ndeltaY*deltaY);

%Forenkling
s_Phi=sin(phi);     c_Phi=cos(phi);
s_Theta=sin(theta); c_Theta=cos(theta); t_Theta=tan(theta);
s_Psi=sin(psi);     c_Psi=cos(psi);

%% System likninger
%Vinkelhastigher roll
dP=1/(I_xx*I_zz-I_xz^2)*(I_zz*Lbar+I_xz*N+(I_yy*I_zz-I_zz^2-I_xz^2)*...
```

```
    Q*R+I_xz*(I_xx-I_yy+I_zz)*P*Q);
%Vinkelhastighet pitch
dQ=1/I_yy*(M+I_xz*(R^2-P^2)+(I_zz-I_xx)*P*R);
%Vinkelhastighet yaw
dR=1/(I_xx*I_zz-I_xz^2)*(I_xz*Lbar+I_xx*N+(I_xz^2-I_xx*I_yy+I_xx^2)*...
    P*Q+I_xz*(I_yy-I_zz-I_xx)*Q*R);

%Eulervinklene
dPhi=P+t_Theta*(Q*s_Phi+R*c_Phi);
dTheta=Q*c_Phi-R*s_Phi;
dPsi=(Q*s_Phi+R*c_Phi)/c_Theta;

%% Lineariserer

%Betingelsene for partiell derivasjon av Roll
fr=[dP,dPhi]';
xr0=[P,phi]';
ur0=deltaR;

%Betingelsene for partiell derivasjon av Pitch
fp=[dQ,dTheta]';
xp0=[Q,theta]';
up0=deltaP;

%Betingelsene for partiell derivasjon av Yaw
fy=[dR,dPsi]';
xy0=[R,psi]';
uy0=deltaY;

disp('  Finner jacobimatrisene for de tre indre sløyfene')
Ar=jacobian(fr,xr0);
Br=jacobian(fr,ur0);
Ap=jacobian(fp,xp0);
Bp=jacobian(fp,up0);
Ay=jacobian(fy,xy0);
By=jacobian(fy,uy0);

%Arbeidspunkt
V_T=200; alpha=5*pi/180; beta=0;
P=0; Q=0; R=0;
phi=0; theta=0; psi=0;
deltaR=0; deltaP=0; deltaY=0;

Ar=double(subs(Ar));
Br=double(subs(Br));

Ap=double(subs(Ap));
Bp=double(subs(Bp));

Ay=double(subs(Ay));
By=double(subs(By));

%% Ser etter om systemene er styrbare
n=length(Ar);
[Ss,Rs]=styrbar(Ar,Br);
if Rs<n
    error('Roll systemet er ikke styrbart R<n')
else
    disp(' Roll systemet er styrbart')
```

```
end

n=length(Ap);
[Ss,Rs]=styrbar(Ap,Bp);
if Rs<n
    error('Pitch systemet er ikke styrbart R<n')
else
    disp(' Pitch systemet er styrbart')
end

n=length(Ay);
[Ss,Rs]=styrbar(Ay,By);
if Rs<n
    error('Yaw systemet er ikke styrbart R<n')
else
    disp(' Yaw systemet er styrbart')
end
```

C.6.3 Styrbar

```
%Styrbarhetsmatrise
function [S,r]=styrbar(A,B)
S=B;
for i=1:(length(A)-1)
    S=[S A^(i)*B];
end
r=rank(S);
```

C.6.4 Indre regulator

```
function [Gr Gp Gy]=IndreReg(Ar, Br, Ap, Bp, Ay, By, i,j)
%Dato: 20.05.10
%Av Thomas Algarheim
%   Indre regulator for flymodell med konste aerodynamiske koeffisienter
%   i=1 sammenfallende poler
%   i=2 Butterworth polynomial
%   j=1 Pitch reg med
%   j=2 Pitch reg er ikke med

%% Reg Roll
Aru=[Ar [0 0]'
     0 -1  0];
Bru=[Br' 0]';

if i==1
   %Sammenfallende poler
    w0=50; %Knekkfrekvensen
    syms sp
    n=2;
    a1 = (sp+w0)^n;
    a2=sym2poly(a1);
    a=conv(a2,[1 w0]);
    regpol=roots(a);
else
    %Butterworth polynomial
    wr0=50; %Knekkfrekvensen
    n=length(Aru);
    alle_poler=roots([(-1)^n, zeros(1,2*n-1),1]);
    regpol=alle_poler(find(real(alle_poler)<0))*wr0;
end

Gr=place(Aru,Bru,regpol);
```

```
%% Reg Pitch
if j==1
    Apu=[Ap [0 0]'
         0 -1  0];
    Bpu=[Bp' 0]';
    if i==1
        %Sammenfallende poler
        w0=20; %Knekkfrekvensen
        syms sp
        n=2;
        a1 = (sp+w0)^n;
        a2=sym2poly(a1);
        a=conv(a2,[1 w0]);
        regpol=roots(a);
    else
        %Butterworthpolynom
        wp0=20; %Knekkfrekvensen
        n=length(Apu);
        alle_poler=roots([(-1)^n, zeros(1,2*n-1),1]);
        regpol=alle_poler(find(real(alle_poler)<0))*wp0;
    end

    Gp=place(Apu,Bpu,regpol);

else
    Gp=0;
    disp(' ')
end

%% Reg Yaw
Ayu=[Ay [0 0]'
     0  -1  0];
Byu=[By' 0]';

if i==1
    %Sammenfallende poler
    w0=30; %Knekkfrekvensen
    syms sp
    n=2;
    a1 = (sp+w0)^n;
    a2=sym2poly(a1);
    a=conv(a2,[1 w0]);
    regpol=roots(a);
else
    %Butterworth polynomial
    wy0=30; %Knekkfrekvensen
    n=length(Ayu);
    alle_poler=roots([(-1)^n, zeros(1,2*n-1),1]);
    regpol=alle_poler(find(real(alle_poler)<0))*wy0;
end

Gy=place(Ayu,Byu,regpol);
```

C.6.5 Kraft og moment på flyet med konstante aerodynamiske koeffisienter

```
function yf = Kraft_Moment(in)

% Beregner kraft og moment på flyet
% Av: Algarheim april 2010
```

```
global rho S c b
global C_D0 C_Dalpha2
global C_Lalpha C_LQ        C_Ldelta1 C_Ldelta2 C_Ldelta3 C_Ldelta4
global C_Cbeta C_CR         C_Cdelta1 C_Cdelta2 C_Cdelta3 C_Cdelta4
global C_lbeta C_lP C_lR    C_ldelta1 C_ldelta2 C_ldelta3 C_ldelta4
global C_m0 C_malpha C_mQ   C_mdelta1 C_mdelta2 C_mdelta3 C_mdelta4
global C_nbeta C_nP C_nR    C_ndelta1 C_ndelta2 C_ndelta3 C_ndelta4

%Split input signal
delta(1) = in(1); %Ror
delta(2) = in(2); %Ror
delta(3) = in(3); %Ror
delta(4) = in(4); %Ror
V_T = in(5);    % Hastighet i x_b -retning
alpha = in(6);  % Angrepsvinkel
beta = in(7);   % Sideslipsvinkel
P = in(11);  % Roll vinkelhastighet
Q = in(12);  % Pitch vinkelhastighet
R = in(13);  % Yaw vinkelhastighet

qbar=0.5*rho*V_T^2; %Dynamisk trykk

%Drag
D=qbar*S*(C_D0+C_Dalpha2*alpha^2);
%Lift
L=qbar*S*(C_Lalpha*alpha+c/V_T*(C_LQ*Q)+...
    C_Ldelta1*delta(1)+C_Ldelta2*delta(2)+C_Ldelta3*delta(3)+...
    C_Ldelta4*delta(4));
%Sidekraft
C=qbar*S*(C_Cbeta*beta+b/V_T*(C_CR*R)+...
    C_Cdelta1*delta(1)+C_Cdelta2*delta(2)+C_Cdelta3*delta(3)+...
    C_Cdelta4*delta(4));
%Rull moment
Lbar=qbar*S*b*(C_lbeta*beta+b/V_T*(C_lP*P+C_lR*R)+...
    C_ldelta1*delta(1)+C_ldelta2*delta(2)+C_ldelta3*delta(3)+...
    C_ldelta4*delta(4));
%Pitc moment
M=qbar*S*c*(C_m0+C_malpha*alpha+c/V_T*(C_mQ*Q)+...
    C_mdelta1*delta(1)+C_mdelta2*delta(2)+C_mdelta3*delta(3)+...
    C_mdelta4*delta(4));
%yawing moment
N=qbar*S*b*(C_nbeta*beta+b/V_T*(C_nP*P+C_nR*R)+...
    C_ndelta1*delta(1)+C_ndelta2*delta(2)+C_ndelta3*delta(3)+...
    C_ndelta4*delta(4));

% Out
yf = [D L C Lbar M N]';
```

C.6.6 Kraft og moment på flyet med variable aerodynamiske koeffisienter

```
function yf = Kraft_Moment_adap(in)

% Beregner kraft og moment på flyet
% Author: Algarheim april 2010

global rho S c b
global C_D0 C_Dalpha2
global C_Lalpha C_LQ        C_Ldelta1 C_Ldelta2 C_Ldelta3 C_Ldelta4
global C_Cbeta C_CR         C_Cdelta1 C_Cdelta2 C_Cdelta3 C_Cdelta4
global C_lbeta C_lP C_lR    C_ldelta1 C_ldelta2 C_ldelta3 C_ldelta4
global C_m0  C_mQ   C_mdelta1 C_mdelta2 C_mdelta3 C_mdelta4
```

```
global C_nbeta C_nP C_nR    C_ndelta1 C_ndelta2 C_ndelta3 C_ndelta4

%Split input signal
delta(1) = in(1);
delta(2) = in(2);
delta(3) = in(3);
delta(4) = in(4);
C_malpha = in(5);
V_T = in(6);    % Hastighet i x_b -retning
alpha = in(7);  % Angrepsvinkel
beta = in(8);   % Sideslipsvinkel
P = in(12);  % Roll vinkelhastighet
Q = in(13);  % Pitch vinkelhastighet
R = in(14);  % Yaw vinkelhastighet

qbar=0.5*rho*V_T^2; %

%Drag
D=qbar*S*(C_D0+C_Dalpha2*alpha^2);
%Lift
L=qbar*S*(C_Lalpha*alpha+c/V_T*(C_LQ*Q)+...
    C_Ldelta1*delta(1)+C_Ldelta2*delta(2)+C_Ldelta3*delta(3)+...
    C_Ldelta4*delta(4));
%Sidekraft
C=qbar*S*(C_Cbeta*beta+b/V_T*(C_CR*R)+...
    C_Cdelta1*delta(1)+C_Cdelta2*delta(2)+C_Cdelta3*delta(3)+...
    C_Cdelta4*delta(4));
%Rull moment
Lbar=qbar*S*b*(C_lbeta*beta+b/V_T*(C_lP*P+C_lR*R)+...
    C_ldelta1*delta(1)+C_ldelta2*delta(2)+C_ldelta3*delta(3)+...
    C_ldelta4*delta(4));
%Pitc moment
M=qbar*S*c*(C_m0+C_malpha*alpha+c/V_T*(C_mQ*Q)+...
    C_mdelta1*delta(1)+C_mdelta2*delta(2)+C_mdelta3*delta(3)+...
    C_mdelta4*delta(4));
%yawing moment
N=qbar*S*b*(C_nbeta*beta+b/V_T*(C_nP*P+C_nR*R)+...
    C_ndelta1*delta(1)+C_ndelta2*delta(2)+C_ndelta3*delta(3)+...
    C_ndelta4*delta(4));

% Out
yf = [D L C Lbar M N]';
```

C.6.7 Systemlikningene til flymodellen

```
function xdot = System_fly(in)
%Dato: 23.04.10
%Av Thomas Algarheim
%   Likninger for fly med kvatrioner

global I_xx I_yy I_zz I_xz m g

%Split input signal
V_T   = in(1);  % Hastighet i vind koordinater
alpha = in(2);  % Angrepsvinkel
beta  = in(3);  % Sideslipsvinkel

%kvatrioner
q0 = in(4);
q1 = in(5);
q2 = in(6);
```

```
q3 = in(7);

P = in(8);       % Roll vinkelhastighet
Q = in(9);       % Pitch vinkelhastighet
R = in(10);      % Yaw vinkelhastighet

% X = in(11);    % X- posisjon
% Y = in(12);    % Y- posisjon
% Z = in(13);    % Z- posisjon

U = in(14);      % Hastighet sett fra fly
V = in(15);      % Hastighet sett fra fly
W = in(16);      % Hastighet sett fra fly

D = in(17);      % Drag
L = in(18);      % Løft
C = in(19);      % Sidekraft

Lbar = in(20);   % Roll moment
M = in(21);      % Pitch moment
N = in(22);      % Yaw moment

T=in(23);
%Forenkling
c_alpha=cos(alpha);c_beta=cos(beta); s_alpha=sin(alpha);s_beta=sin(beta);
t_beta=tan(beta);
%Krefter på fly
F_x=-D+T;F_y=-C;F_z=-L;

xdot = zeros(16,1);
%Vindhastighet
xdot(1)=1/m*(T*cos(alpha)*cos(beta)-D+m*...
    g*(cos(beta)*sin(alpha)*(q0^2-q1^2-q2^2+q3^2)-sin(beta)*...
    (2*q0*q1-2*q2*q3)+cos(alpha)*cos(beta)*(2*q0*q2+2*q1*q3)));
%Angrepsvinkel
xdot(2)=-P*c_alpha*t_beta+Q-R*s_alpha*t_beta+...
    1/(m*V_T*c_beta)*(T*s_alpha-L+m*g*(s_alpha*(2*q0*q2 + 2*q1*q3)-...
    c_alpha*(q0^2-q1^2-q2^2+q3^2)));
%Sideslipsvinkel
xdot(3)=P*s_alpha-R*c_alpha+1/(m*V_T)*(-T*c_alpha*s_beta-C-m*g*(c_beta*...
    (2*q0*q1-2*q2*q3)+s_alpha*s_beta*(q0^2-q1^2-q2^2+q3^2)+c_alpha*...
    s_beta*(2*q0*q2+2*q1*q3)));

%kinematikk
    %Euler parametrene
xdot(4)=0.5*(-q1*P-q2*Q-q3*R);  %q0
xdot(5)=0.5*(q0*P-q3*Q+q2*R);   %q1
xdot(6)=0.5*(q3*P+q0*Q-q1*R);   %q2
xdot(7)=0.5*(-q2*P+q1*Q+q0*R);  %q3

%Vinkelakselerasjoner i flykoordinater
xdot(8)=1/(I_xx*I_zz-I_xz^2)*(I_zz*Lbar+I_xz*N+(I_yy*I_zz-I_zz^2-...
    I_xz^2)*Q*R+I_xz*(I_xx-I_yy+I_zz)*P*Q);         %Roll
xdot(9)=1/I_yy*(M+I_xz*(R^2-P^2)+(I_zz-I_xx)*P*R);  %Pitch
xdot(10)=1/(I_xx*I_zz-I_xz^2)*(I_xz*Lbar+I_xx*N+(I_xz^2-
I_xx*I_yy+I_xx^2)*...
    P*Q+I_xz*(I_yy-I_zz-I_xx)*Q*R);                 %Yaw

%Hastigheter sett fra jordkoordinatsystemet
```

```
xdot(11)=V_T*(s_beta*(2*q0*q3 + 2*q1*q2) + c_alpha*c_beta*...
    (q0^2 + q1^2 - q2^2 - q3^2) - c_beta*s_alpha*(2*q0*q2 - 2*q1*q3));% x
xdot(12)=V_T*(s_beta*(q0^2 - q1^2 + q2^2 - q3^2) - c_alpha*...
    c_beta*(2*q0*q3 - 2*q1*q2) + c_beta*s_alpha*(2*q0*q1 + 2*q2*q3)); % y
xdot(13)=V_T*(c_beta*s_alpha*(q0^2 - q1^2 - q2^2 + q3^2) - ...
    s_beta*(2*q0*q1 - 2*q2*q3) + c_alpha*c_beta*(2*q0*q2 + 2*q1*q3)); % z

%Akselerasjoner sett fra flykoordinatsystemet
xdot(14)=(F_x/m)-Q*W+R*V+g*(2*q0*q2 + 2*q1*q3);          %U
xdot(15)=(F_y/m)-U*R+W*P-g*(2*q0*q1 - 2*q2*q3);          %V
xdot(16)=(F_z/m)-V*P+Q*U+g*(q0^2-q1^2-q2^2+q3^2);        %W
```

C.6.8 Kvatrioner til Eulervinkler

```
function out = q2e( in )
%Dato: 23.04.10
%Av Thomas Algarheim
%   Kvatrioner til eulervinkler
q0=in(1);
q1=in(2);
q2=in(3);
q3=in(4);
out=[...
    atan2(2*(q0*q1+q2*q3),1-2*(q1^2+q2^2))      % Roll
    asin(2*(q0*q2-q3*q1))                       % Pitch
    atan2(2*(q0*q3+q1*q2),1-2*(q2^2+q3^2))];    % Yaw
end
```

C.6.9 Ytre regulator

```
function [Kp Ti Td Tf]=YtreReg(Kpk,Tp,i)
%Dato: 20.05.10
%Av Thomas Algarheim
%   Zegler Nicels innstiling av PID regulatorer
%   i=1 P-  regulator
%   i=2 PI- regulator
%   i=3 PID-regulator
%   i=4 For innstilling Kpk=Kp

if i==1
    Kp=Kpk*0.5;
    Ti=0;
    Td=0;
    Tf=0.1*Td;
elseif i==2
    Kp=Kpk*0.45;
    Ti=Tp/1.2;
    Td=0;
    Tf=0.1*Td;
elseif i==3
    Kp=Kpk*0.6;
    Ti=Tp/2;
    Td=Tp/8;
    Tf=0.1*Td;
else
    Kp=Kpk;
    Ti=0;
    Td=0;
    Tf=0.1*Td;
end
```

C.6.10 Bane

```
function [x_p y_p z_p t] = Bane(wpt_x, wpt_y, wpt_z, wpt_t, ts, time)
%Dato: 23.04.10
%Av Thomas Algarheim
%   Gi referansepunkter og du får banen som passer

% Eksempel
% ts=0.01; %Sampel tiden
% time=20;
% disp('Finner bane ut i fra referanse punkter')
% wpt_x = [0 100 200 600  1000  2000  3600    4000]; %ca 200 m/s
% wpt_y = [0 0    0   5    0     0     0       0];
% wpt_z = [0 0    5   0    0     0     0       0];
% wpt_t = [0 0.5  1   3    5     10    18      20];
%

t=0:ts:time;
x_p = pchip(wpt_t,wpt_x,t);
y_p = pchip(wpt_t,wpt_y,t);
z_p = pchip(wpt_t,wpt_z,t);
```

C.6.11 Vektor velger

```
function y=vektor_velger(in)
%Dato: 23.04.10
%Av Thomas Algarheim
%   Finner referansen til den bestemte tiden, ut i fra en vektor

global Ts
t=0:Ts:in(1);
y=in(length(t+1));
```

C.6.12 Dtf

```
function [Az,Bz]=Dtf(V_T,C_malpha, Ts)
%Dato: 19.5.10
%Av Thomas Algarheim
% Finner den diskrete transferfunksjonen av den forenklede pitch systemet
% ved hjelp av rekkeutvikling

global m I_yy rho S c C_mdeltaP C_Lalpha C_mQ C_LdeltaP

K1=0.5*rho*S*V_T^2*c*C_mdeltaP;
K2=0.5*rho*S*V_T^2*c*C_malpha;
K3=1/(2*m)*rho*S*V_T*(C_Lalpha);
K4=0.25*rho*S*V_T*c^2*C_mQ;%1/4*rho*s*v*c^2*Cmomega
K5=1/(2*m)*rho*S*V_T*(C_LdeltaP);

Ap=[-(K3)      K3      0;
    0        0       1;
    -(K2)/I_yy (K2)/I_yy   K4/I_yy]; %forkorterbort A(3,3) for denne har
lite å si  K4/Iyy
Bp=[K5       0       K1/I_yy]';
Cp=[0 1 0];
Dp=0;

Ap=double(subs(Ap));
Bp=double(subs(Bp));

% Phi=expm(Ap*Ts);
% syms x
```

```
% %Superposisjonsintegralet gir La=int(Fi(t_(k+1),tau)*L(tau) d(tau)
% Delta=real(double(int(expm(Ap*(Ts-x))*Bp,x,0,Ts)));

%Bruker rekkeutvikling og finner det diskrete systemet
Phi=zeros(size(Ap));

%Hjelpevariabler
A_Phi=Ap*Ts;
Fi_2=zeros(size(Ap));
F_t1=eye(size(Ap));
F_t2=eye(size(Ap));
k=1;
n=10000;
for j=1:n
    Phi=Phi+F_t1;
    Fi_2=Fi_2+F_t2;
    F_t1=A_Phi*F_t1/k;
    F_t2=A_Phi*F_t2/(k+1);
    k=k+1;
end
Delta=Fi_2*Bp*Ts;

b0=-(-Cp(2)*Delta(2));

b1=-(Cp(2)*(Delta(2)*Phi(1,1)+Delta(2)*Phi(3,3)-Delta(1)*Phi(2,1)-...
    Delta(3)*Phi(2,3)));

b2=-(Cp(2)*(...
   Delta(1)*(Phi(2,1)*Phi(3,3)-Phi(3,1)*Phi(2,3))+...
   Delta(2)*(Phi(1,3)*Phi(3,1)-Phi(1,1)*Phi(3,3))+...
   Delta(3)*(Phi(1,1)*Phi(2,3)-Phi(2,1)*Phi(1,3))));

a1=-(Phi(1,1)+Phi(2,2)+Phi(3,3));

a2=-(Phi(1,2)*Phi(2,1)+Phi(1,3)*Phi(3,1)+Phi(2,3)*Phi(3,2)...
   -Phi(1,1)*(Phi(2,2)+Phi(3,3))-Phi(2,2)*Phi(3,3));

a3=-(Phi(1,1)*(Phi(2,2)*Phi(3,3)-Phi(2,3)*Phi(3,2))...
  +Phi(1,2)*(Phi(3,1)*Phi(2,3)-Phi(2,1)*Phi(3,3))...
  +Phi(1,3)*(Phi(2,1)*Phi(3,2)-Phi(2,2)*Phi(3,1)));
Bz=[b0 b1 b2]';
Az=[1 a1 a2 a3]';
```

C.6.13 Pitch Regulator

```
function [F G H]=PitchReg(Az,Bz,Tp)
%Dato: 19.5.10
%Av Thomas Algarheim
% Bruker poltildeligsmetoden til å finne regulatorpolynomene F,G og H

t1=Tp(2);t2=Tp(3);t3=Tp(4);

n_a=3;
n_b=2;
n_f=n_b;
n_g=n_a-1;

a1=Az(2); a2=Az(3); a3=Az(4);
b0=Bz(1); b1=Bz(2); b2=Bz(3);
```

```
H=(1+t1+t2+t3)/(b0+b1+b2);%1/(b0+b1+b2)

Ar=[1  0   b0 0   0  ;    %[f1]
    a1 1   b1 b0  0  ;    %[f2]
    a2 a1  b2 b1  b0 ;    %[g0]
    a3 a2  0  b2  b1 ;    %[g1]
    0  a3  0  0   b2];    %[g2]
Br=[(t1-a1) (t2-a2) (t3-a3) 0 0]';

theta_r=Ar\Br; %[f1 f2 g0 g1 g2]'

F=[1 theta_r(1:n_f)'];
G=theta_r(n_f+1:length(theta_r))';
```